Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа по дисциплине «Защита информации»

Тема Алгоритм шифрования DES и режимы шифрования

Студент Светличная А.А.

Группа ИУ7-73Б

Преподаватель Чиж И. С.

Москва — 2023 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Шифрование информации является ключевым элементом обеспечения безопасности данных в современном информационном обществе. В этом контексте алгоритм Data Encryption Standard (DES) занимает особое место, представляя собой один из первых стандартов шифрования, который оказал значительное влияние на развитие криптографии. Разработанный в начале 1970-х годов, DES стал неотъемлемой частью информационной безопасности, нашедшей широкое применение в коммерческих, государственных и банковских системах.

Цель: разработка программной реализации шифровального алгоритма DES в режиме работы по варианту.

Задачи:

- исследование исторических аспектов данного алгоритма;
- анализ алгоритма DES;
- программная реализация данного алгоритма.

1 Аналитическая часть

Историческая справка

DES, или Data Encryption Standard, представляет собой симметричный алгоритм шифрования, который был разработан в начале 1970-х годов в США. Этот алгоритм является результатом усилий Национального института стандартов и технологии (NIST) и Национальной агентство стандартов (NSA) с целью создания стандарта для шифрования данных в правительственных и коммерческих системах [1].

Начало разработки (начало 1970-х годов): DES был разработан командой криптографов под руководством IBM. В 1973 году Национальный институт стандартов и технологии (NIST) объявил конкурс на разработку стандарта шифрования данных, предназначенного для использования в федеральных информационных системах [1].

Выбор DES (1977 год): Алгоритм, предложенный IBM, стал победителем конкурса. DES был выбран как стандарт шифрования и был опубликован в документе под названием "Data Encryption Standard"в январе 1977 года [1].

Коммерческое и широкое использование: DES стал широко применяться в банковских, коммерческих и государственных системах. Он служил основой для многих безопасных протоколов, таких как SSL (Secure Sockets Layer) и TLS (Transport Layer Security) [1].

Критика и выход за пределы (1990-е годы): С течением времени вычислительные мощности увеличивались, и DES стал подвергаться критике из-за сравнительно короткой длины ключа. В 1999 году EFF (Electronic Frontier Foundation) использовала распределенные вычислительные ресурсы для успешного взлома DES-ключа [1].

Замена стандарта (2001 год): В результате критики и изменений в технологическом ландшафте NIST объявил, что DES больше не является безопасным стандартом для шифрования. Он был заменен более современными алгоритмами, такими как AES (Advanced Encryption Standard) [1].

Хотя DES устарел, и его использование сейчас не рекомендуется из-за относительной слабости, его история важна, так как он является одним из первых стандартов шифрования, который применялся на широком уровне.

2 Конструкторская часть

2.1 Шифровальный алгоритм DES

Процесс шифрования заключается в начальной перестановке битов 64-битового блока, шестнадцати циклах шифрования и обратной перестановки битов [2]. Схемы алгоритма показаны на рисунках 2.1 и 2.2 (более подробная).

Рисунок 2.1 – Обобщенная схема шифрования в алгоритме DES

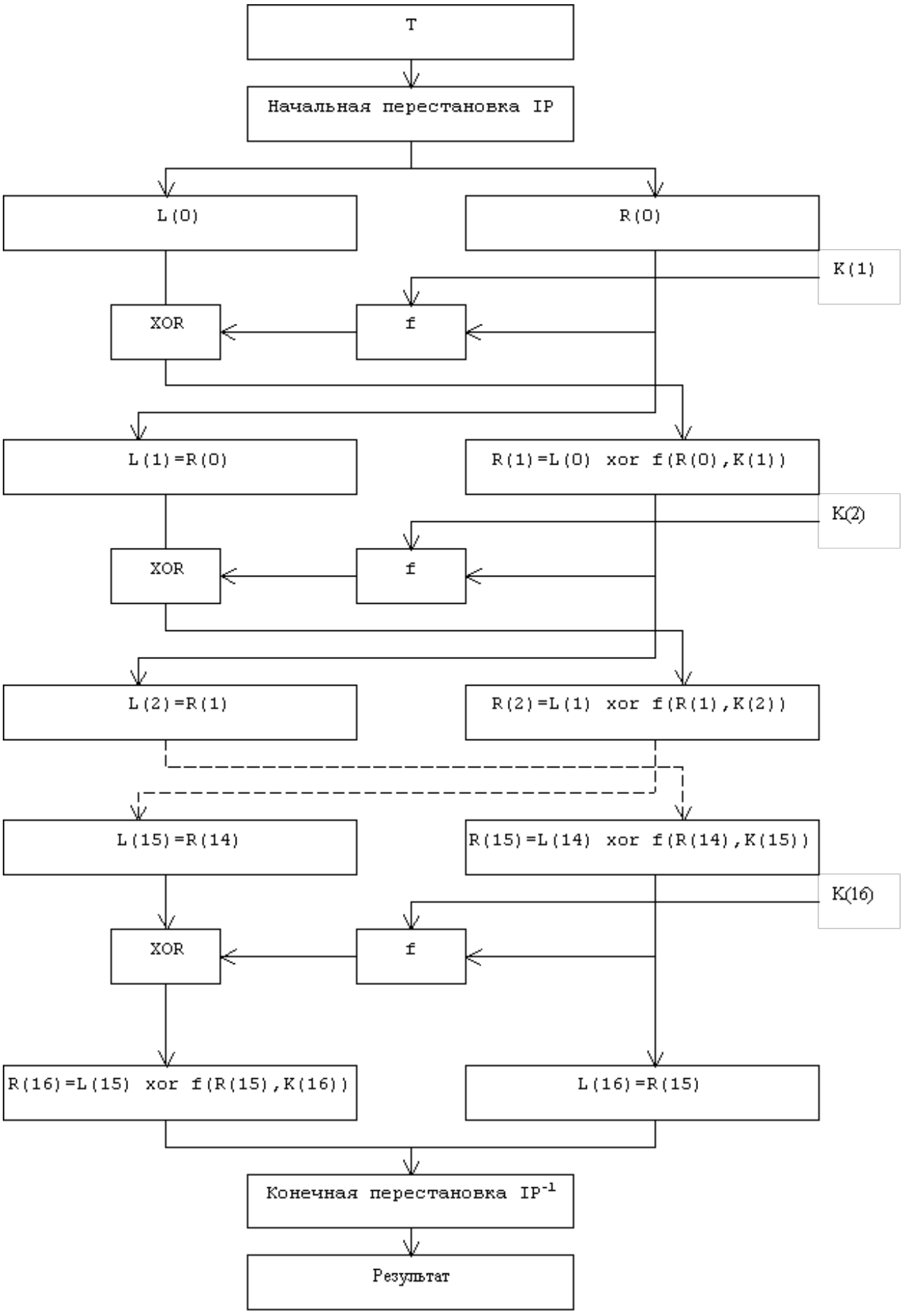

Рисунок 2.2 – Структура алгоритма шифрования DES

В данных схемах вводятся ранее неупомянутые функция f и ключи, схемы работы с ними представлена на рисунках 2.3 и 2.4.

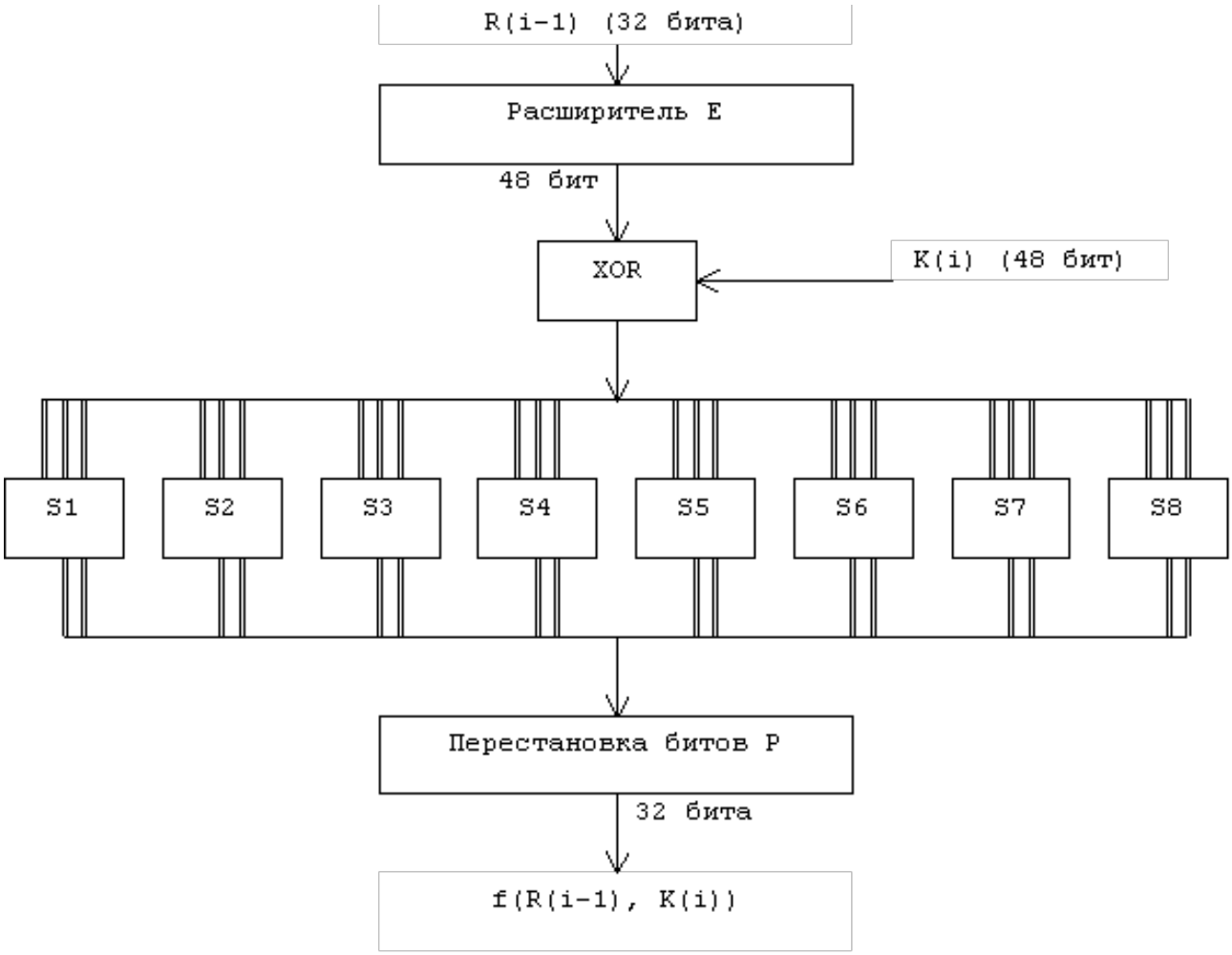

Рисунок 2.3 – Вычисление функции f(R(i-1), K(i))

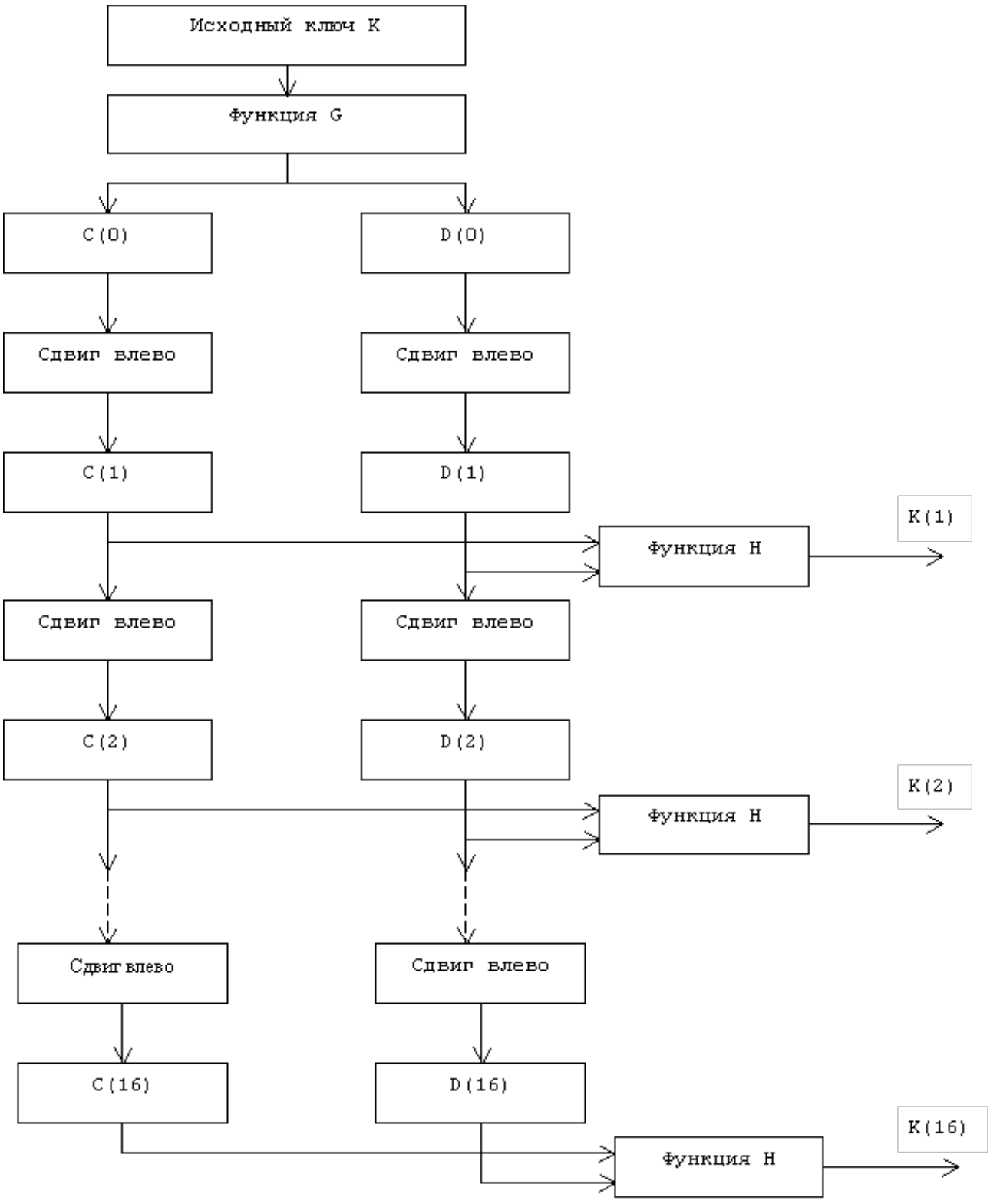

Рисунок 2.4 – Вычисление ключа K(i)

Преобразования выполняются с помощью стандартных таблиц (8 таблиц), суть работы которых заключается втом, что бит, который ранее был на n-ой позиции становится теперь битом с позицией m, где n - значение в таблице, m - номер данного значения в таблице. Иная схема используется только в таблице сдвигов и функции преобразования f. Функция преобразования ра-

ботает следующим образом: пусть на вход функции-матрицы S_j поступает 6-битовый блок $B(j) = b_1b_2b_3b_4b_5b_6$. Тогда двухбитовое число b_1b_6 указывает номер строки матрицы, а $b_2b_3b_4b_5$ - номер столбца. Результатом $S_j(B(j))$ будет 4-битовый элемент, расположенный на пересечении указанных строки и столбца [2].

2.2 Режим работы

Чтобы воспользоваться алгоритмом DES для решения разнообразных криптографических задач, разработаны четыре рабочих режима:

1. Электронная кодовая книга (ECB - Electronic Code Book).
2. Сцепление блоков шифра (CBC - Cipher Block Chaining).
3. Обратная связь по шифртексту (CFB - Cipher Feed Back).
4. Обратная связь по выходу (OFB - Output Feed Back).

На рисунках 2.5 и 2.6 показана схема шифрования и расшифровки для режима OFB.

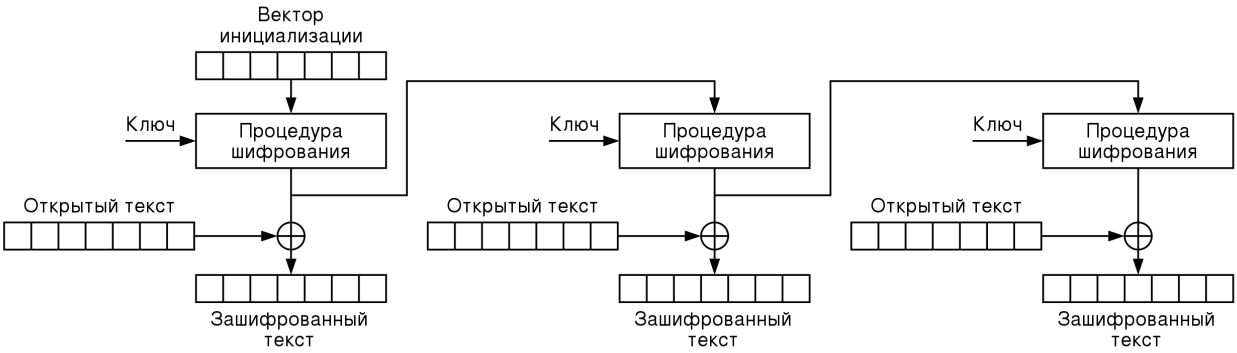

Рисунок 2.5 – Схема шифрования алгоритма DES в режиме OFB

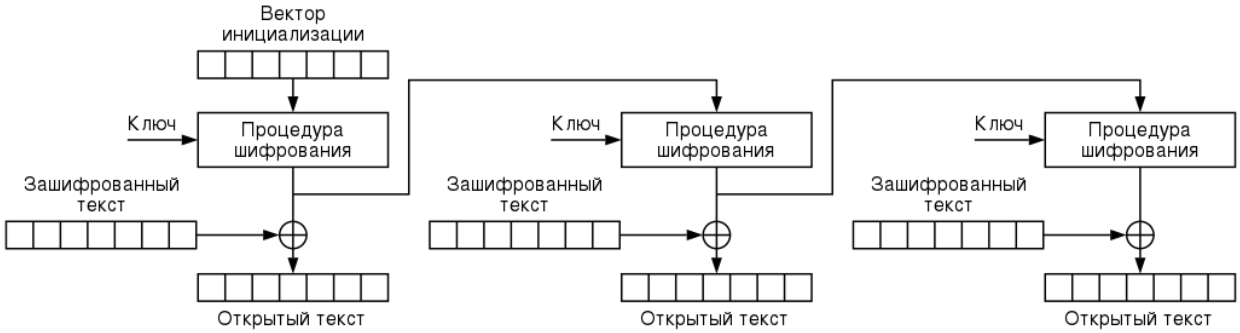

Рисунок 2.6 – Схема расшифровки алгоритма DES в режиме OFB

Нетрудно заметить, что в данном случае шифрование и расшифровка ничем не отличаются друг от друга.

3 Технологическая часть

3.1 Реализация алгоритма

Листинг 3.1 – Реализация части работы с ключами алгоритма DES

```
1  int key64to56(int pos, int text)
2  {
3      for (int i = 0; i < 56; i++)
4          if (PC1[i] == pos + 1)
5              return key56bit[i] = text;
6  }
7
8  int key56to48(int round, int pos, int text)
9  {
10     for (int i = 0; i < 56; i++)
11         if (PC2[i] == pos + 1)
12             return key48bit[round][i] = text;
13 }
14
15 void key64to48(unsigned int key[])
16 {
17     int k, backup[17][2];
18     int CD[17][56];
19     int C[17][28], D[17][28];
20
21     for (int i = 0; i < 64; i++)
22         key64to56(i, key[i]);
23
24         for (int i = 0; i < 28; i++)
25                 C[0][i] = key56bit[i];
26         for (int i = 28; i < 56; i++)
27                 D[0][i - 28] = key56bit[i];
28
29
30
```

```
31      for (int i = 1; i < 17; i++)
32      {
33          int shift = SHIFTS[i - 1];
34
35          for (int j = 0; j < shift; j++)
36              backup[i - 1][j] = C[i - 1][j];
37          for (int j = 0; j < (28 - shift); j++)
38              CD[i][j] = C[i][j] = C[i - 1][j + shift];
39          for (int j = 28 - shift, k = 0; j < 28; j++)
40              CD[i][j] = C[i][j] = backup[i - 1][k++];
41
42          for (int j = 0; j < shift; j++)
43              backup[i - 1][j] = D[i - 1][j];
44          for (int j = 0; j < (28 - shift); j++)
45              CD[i][28 + j] = D[i][j] = D[i - 1][j + shift];
46          for (int j = 28 - shift, k = 0; j < 28; j++)
47              CD[i][28 + j] = D[i][j] = backup[i - 1][k++];
48      }
49
50      for (int i = 1; i < 17; i++)
51          for (int j = 0; j < 56; j++)
52              key56to48(i, j, CD[i][j]);
53  }
```

Листинг 3.2 – Реализация шифрования/расшифровки алгоритма DES в режиме OFB

```
int initialPermutation(int pos, int text)
{
    for (int i = 0; i < 64; i++)
        if (IP[i] == pos + 1)
            return IPtext[i] = text;;
}

void expansionFunction(int pos, int text)
{
    for (int i = 0; i < 48; i++)
        if (E[i] == pos + 1)
            EXPtext[i] = text;
}

int XOR(int a, int b) {
    return (a ^ b);
}

int F1(int i)
{
    int r, c, b[6];

    for (int j = 0; j < 6; j++)
        b[j] = X[i][j];

    r = b[0] * 2 + b[5];
    c = 8 * b[1] + 4 * b[2] + 2 * b[3] + b[4];

    return S[i][r][c];
}


```

```
33 int SBox(int XORtext[])
34 {
35     int k = 0;
36     for (int i = 0; i < 8; i++)
37         for (int j = 0; j < 6; j++)
38             X[i][j] = XORtext[k++];
39 
40     for (int i = 0; i < 8; i++)
41         convertIntToBits(F1(i));
42 }
43 
44 int PBox(int pos, int text)
45 {
46     int i;
47     for (i = 0; i < 32; i++)
48     {
49         if (P[i] == pos + 1) {
50             break;
51         }
52     }
53     R[i] = text;
54 }
55 
56 void cipher(int round)
57 {
58     for (int i = 0; i < 32; i++)
59         expansionFunction(i, RIGHT[round - 1][i]);
60 
61     for (int i = 0; i < 48; i++)
62         XORtext[i] = XOR(EXPtext[i], key48bit[round][i]);
63 
64     SBox(XORtext);
65 
66 
```

```
67      for (int i = 0; i < 32; i++)
68          PBox(i, X2[i]);
69
70      for (int i = 0; i < 32; i++)
71          RIGHT[round][i] = XOR(LEFT[round - 1][i], R[i]);
72  }
73
74  int finalPermutation(int pos, int text)
75  {
76      for (int i = 0; i < 64; i++)
77          if (FP[i] == pos + 1)
78              return ENCRYPTED[i] = text;
79  }
80
81  void Encryption(int plain[])
82  {
83      for (int i = 0; i < 64; i++)
84          initialPermutation(i, plain[i]);
85
86      for (int i = 0; i < 32; i++)
87          LEFT[0][i] = IPtext[i];
88      for (int i = 32; i < 64; i++)
89          RIGHT[0][i - 32] = IPtext[i];
90
91      for (int i = 1; i < 17; i++)
92      {
93          cipher(i, 0);
94
95          for (int j = 0; j < 32; j++)
96              LEFT[i][j] = RIGHT[i - 1][j];
97      }
98
99
100
```

```
101     for (int i = 0; i < 64; i++)
102     {
103         if (i < 32)
104             CIPHER[i] = RIGHT[16][i];
105         else
106             CIPHER[i] = LEFT[16][i - 32];
107         finalPermutation(i, CIPHER[i]);
108     }
109
110     for (int i = 0; i < 64; i++)
111         IV[i] = ENCRYPTED[i];
112 }
113
114 void encrypt(int num_block)
115 {
116     for (int i = 0; i < num_block; i++)
117    {
118         Encryption(IV);
119
120         for (int j = 0; j < 64; j++)
121                 XorTextIV[j] = XOR(plain[j], IV[j]);
122    }
123 }
```

В некоторых функция опущены операции записи или чтения и другие, не относящиеся напрямую к шифру.

3.2 Тестирование

Негативные:

1. Строка запуска: ./app.exe
   Код возврата: 1
   Описание: не задано имя входного файла

2. Строка запуска: ./app.exe noexist.txt
   Код возврата: 2

Описание: не существует входной файл

Позитивные:

1. Строка запуска: ./app.exe input.txt
   key:
   00010011001101000101011101111001100110111011110011011111111110001
   IV:
   01010101111010101010010101100100011100101000101001001010100101 01
   input: a
   input_bits: 01100001
   encrypt_bits:
   00001000110010111111001000110001110100110111001000110110100011 00
   decrypt_bits:
   01100001000000000000000000000000000000000000000000000000000000 00
   decrypt: a
   Код возврата: 0
   Описание: сообщение размером 1 байт

2. Строка запуска: ./app.exe input.zip
   key:
   00010011001101000101011101111001100110111011110011011111111110001
   IV:
   01010101111010101010010101100100011100101000101001001010100101 01
   input: a
   input_bits: 01100001
   encrypt_bits:
   00001000110010111111001000110001110100110111001000110110100011 00
   decrypt_bits:
   01100001000000000000000000000000000000000000000000000000000000 00
   decrypt: a
   Код возврата: 0
   Описание: входной файл является архивом

3. Строка запуска: ./app.exe input.txt
   key:

0001001100110100010101110111100110011011101111001101111111110001
IV:
0101010111101010101001010110010001110010100010100100101010010101
input: aaaaaaaaa
decrypt: aaaaaaaaa
Код возврата: 0
Описание: входное сообщение более 64 бит

ЗАКЛЮЧЕНИЕ

В ходе лабораторной работы была достигнута поставленная **цель**: разработана программная реализации шифровального алгоритма DES в режиме работы по варианту.

Все **задачи** лабораторной работы выполнены:

- исследованы исторических аспектов данной алгоритма;
- проведен анализ алгоритма DES;
- программно реализован данный алгоритм.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. DES: История стандарта шифрования данных [Электронный ресурс]. URL: https://coinrivet.com/ru/des-the-story-of-the-data-encryption-standard/ (дата обращения: 13.10.2023).

2. Алгоритм DES [Электронный ресурс]. URL: https://www.opennet.ru/docs/RUS/inet_book/6/des_641.html (дата обращения: 13.10.2023).