



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»

## Отчёт по лабораторной работе №3 по курсу «Защита информации» вариант 7

Тема Симметричный алгоритм AES

Студент Карпова Е.О.

Группа ИУ7-72Б

Преподаватель Чиж И. С.

Москва — 2023 г.

# СОДЕРЖАНИЕ

<b>ВВЕДЕНИЕ</b>	<b>3</b>
<b>1 Аналитический раздел</b>	<b>4</b>
1.1 Стандарт шифрования данных AES . . . . .	4
1.2 Шифрование . . . . .	4
1.3 Расшифрование . . . . .	8
1.4 Режим OFB (Output Feedback) . . . . .	10
<b>2 Конструкторский раздел</b>	<b>12</b>
2.1 Алгоритмы AES . . . . .	12
<b>3 Технологический раздел</b>	<b>14</b>
3.1 Реализация алгоритмов . . . . .	14
3.2 Тестирование . . . . .	15
<b>ЗАКЛЮЧЕНИЕ</b>	<b>17</b>

## ВВЕДЕНИЕ

AES — симметричный алгоритм блочного шифрования, принятый в качестве стандарта шифрования правительством США по результатам конкурса AES. Этот алгоритм хорошо проанализирован и сейчас широко используется, как это было с его предшественником DES. Национальный институт стандартов и технологий США опубликовал спецификацию AES 26 ноября 2001 года после пятилетнего периода, в ходе которого были созданы и оценены 15 кандидатур. 26 мая 2002 года AES был объявлен стандартом шифрования. По состоянию на 2009 год AES является одним из самых распространённых алгоритмов симметричного шифрования.

**Целью** данной лабораторной работы является разработка программного обеспечения, позволяющего шифровать и дешифровать произвольный файл по алгоритму симметричного шифрования AES, а именно OFB — режим обратной связи по выходу. Для достижения цели необходимо решить следующие задачи:

- изучить алгоритм симметричного шифрования AES;
- реализовать алгоритм симметричного шифрования AES;
- реализовать алгоритм режим OFB.

# 1 Аналитический раздел

## 1.1 Стандарт шифрования данных AES

AES представляет собой алгоритм шифрования 128-битных блоков данных ключами по 128, 192 и 256 бит. AES является упрощенной версией алгоритма Rijndael. Оригинальный алгоритм Rijndael отличается тем, что поддерживает более широкий набор длин блоков.

Введем следующие термины.

- *Слово* — последовательность из 4-х байт.
- *Форма* — матрица 4x4 байт, используемая для представления блока.

b0	b4	b8	b12
b1	b5	b9	b13
b2	b6	b10	b14
b3	b7	b11	b15

- *Раунд* — название итерации в цикле преобразований форм.
- $N_b$  — количество слов в блоке.
- $N_k$  — количество слов в ключе.
- $N_r$  — количество раундов.

Количество слов в ключе и количество раундов соотносятся как:

$N_k$	$N_r$
4	10
6	12
8	14

## 1.2 Шифрование

В алгоритме AES применяются следующие преобразования данных:

- 1) *ExpandKey* — вычисление раундных ключей для всех раундов.
- 2) *SubBytes* — подстановка байтов с помощью таблицы подстановок.
- 3) *ShiftRows* — циклический сдвиг строк в форме на различные величины.
- 4) *MixColumns* — смешивание данных внутри каждого столбца формы.
- 5) *AddRoundKey* — сложение ключа раунда с формой.

## Преобразование SubBytes

Преобразование SubBytes заключается в замене каждого байта  $xu$  формы (где  $x$  и  $y$  обозначают шестнадцатеричные цифры) на другой в соответствии таблицей, представленной на рисунке 1.1. Например, байт  $fe$  заменится на  $bb$ .

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	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

Рисунок 1.1 – Таблица подстановок

## Преобразование ShiftRows

Преобразование ShiftRows заключается в циклическом сдвиге влево строк формы. Преобразование схематично представлено на рисунке 1.2. Первая строка остается неизменной. Во второй производится сдвиг на 1 байт, то есть первый байт переносится в конец. В третьей — сдвиг на 2 байта, в четвертой — на 3.

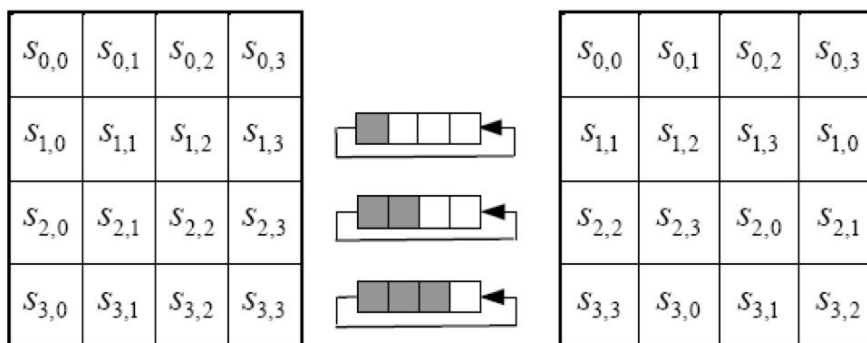


Рисунок 1.2 – Преобразование ShiftRows

## Преобразование MixColumns

Преобразование MixColumns заключается в умножении квадратной матрицы 4-го порядка на каждый столбец формы по следующей формуле:

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

Умножение производится в поле Галуа  $GF(2^8)$ . Над каждым столбцом операция производится отдельно, как показано на рисунке 1.3.

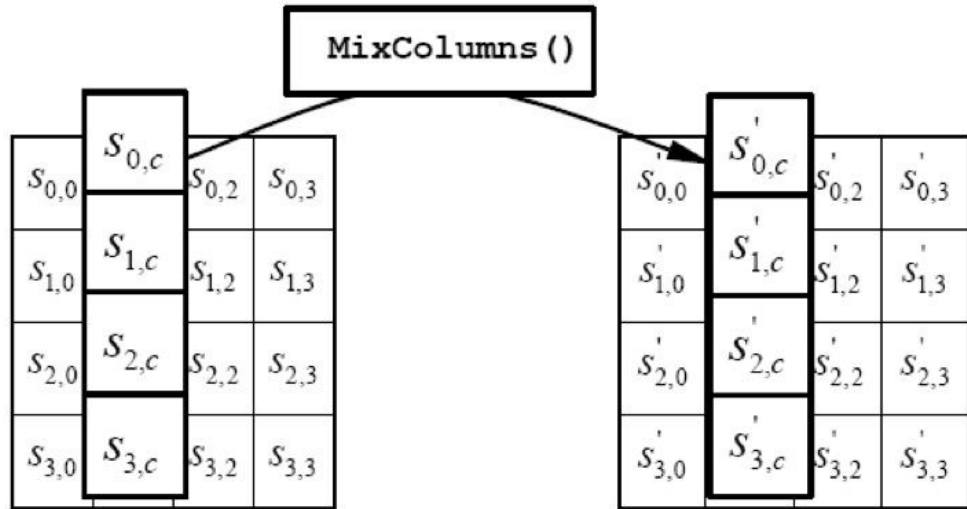


Рисунок 1.3 – Преобразование MixColumns

## Преобразование AddRoundKey

В преобразовании AddRoundKey 32-битные слова раундного ключа прибавляются к столбцам формы с помощью побитовой операции XOR:

$$[s'_{0,c}, s'_{1,c}, s'_{2,c}, s'_{3,c}] = [s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c}] \oplus [w_{round*Nb+c}]$$

Здесь  $w_i$  — это столбцы ключа.

Над каждым столбцом операция производится отдельно, как показано на рисунке 1.4.

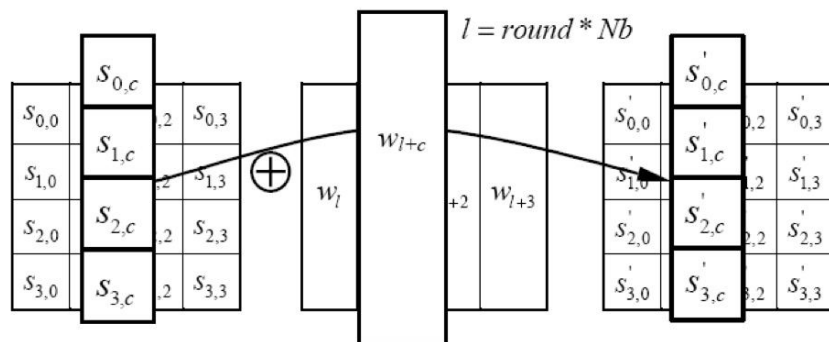


Рисунок 1.4 – Преобразование AddRoundKey

## Процедура ExpandKey

В алгоритме AES генерируются раундные ключи на основе ключа шифрования с помощью процедуры ExpandKey. Процедура ExpandKey создает  $N_b \cdot (N_r + 1)$  слов: алгоритму требуется начальный ключ размером  $N_b$ , плюс каждый из  $N_r$  раундов требует ключ из  $N_b$  слов.

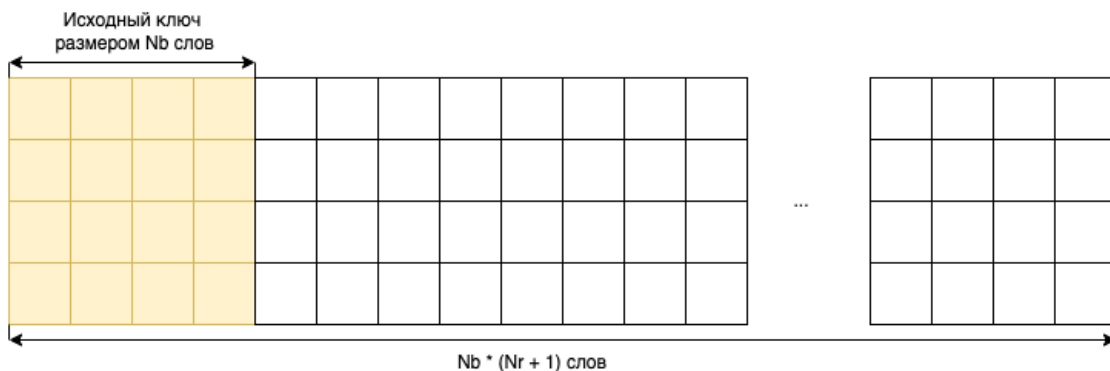


Рисунок 1.5 – Раундовый ключ

Для слов на позициях кратных 4-м ( $w_4, w_8, \dots$ ) используется следующий алгоритм:

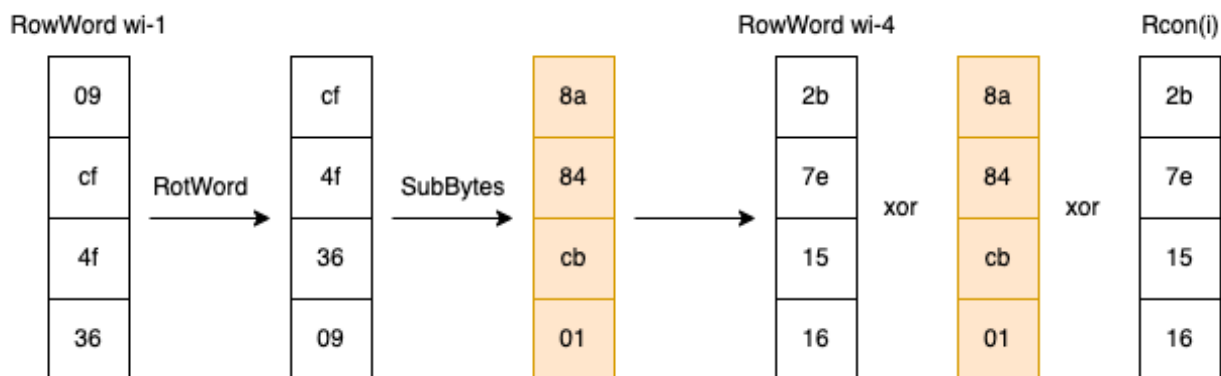


Рисунок 1.6 – Алгоритм для слов кратных 4-м

Для остальных используется следующий алгоритм:

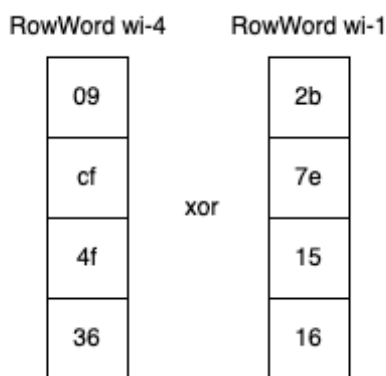


Рисунок 1.7 – Алгоритм для остальных слов

Здесь функция SubWord осуществляет замену каждого байта в слове в соответствии с таблицей подстановок. Функция RotWord осуществляет циклический сдвиг байтов в слове влево.

Функция  $Rcon(i)$  формирует слово  $[02^{i-1}, 00, 00, 00]$ .

### 1.3 Расшифрование

При расшифровании все преобразования производятся в обратном порядке. Используются следующие обратные преобразования вместо соответствующих шифрующих:

- 1) InvSubBytes — подстановка байтов с помощью обратной таблицы подстановок;
- 2) InvShiftRows — циклический сдвиг строк в форме на различные величины;
- 3) InvMixColumns — смешивание данных внутри каждого столбца формы.



Процедуры `ExpandKey` и `AddRoundKey` остаются неизменными. Ключи раунда используются в обратном порядке. Алгоритм расшифрования представлен на рисунке 2.2.

## Преобразование `InvShiftRows`

Данное преобразование обратное преобразованию `ShiftRows`. Схематично преобразование показано на рисунке 1.8. Первая строка формы остается неизменной. Вторая строка циклически сдвигается вправо на 1 байт. Третья — на 2, четвертая — на 3.

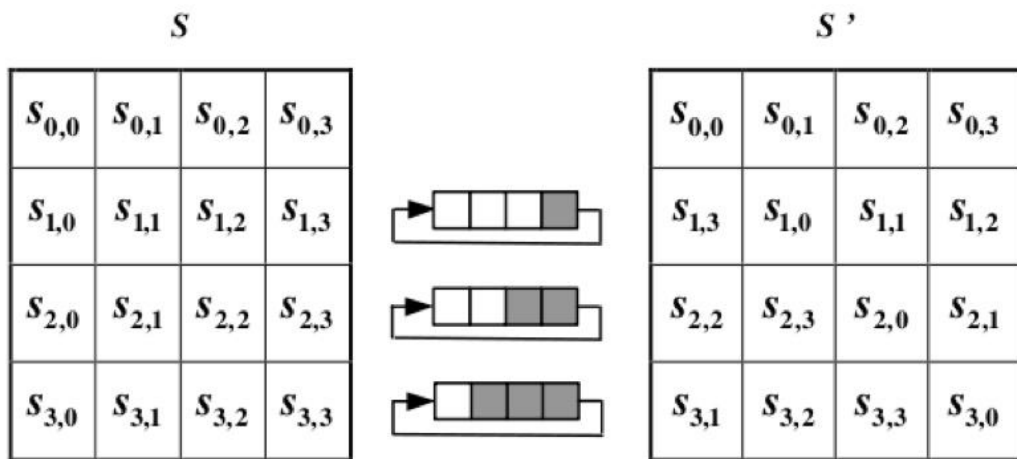


Рисунок 1.8 – Преобразование `InvShiftRows`

## Преобразование `InvSubBytes`

Данное преобразование обратное преобразованию `SubBytes`. Подстановка байтов происходит аналогично с помощью обратной таблицы подстановок, представленной на рисунке 1.9.

		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

Рисунок 1.9 – Обратная таблица подстановок

## Преобразование InvMixColumns

Данное преобразование обратно преобразованию MixColumns. InvMixColumns преобразует в форме каждый столбец отдельно по следующей формуле:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Здесь умножение также производится в поле Галуа  $GF(2^8)$ .

## 1.4 Режим OFB (Output Feedback)

В режиме OFB входным блоком служит результат применения AES к предыдущему входному блоку. Первым входным блоком служит Initialization Vector.

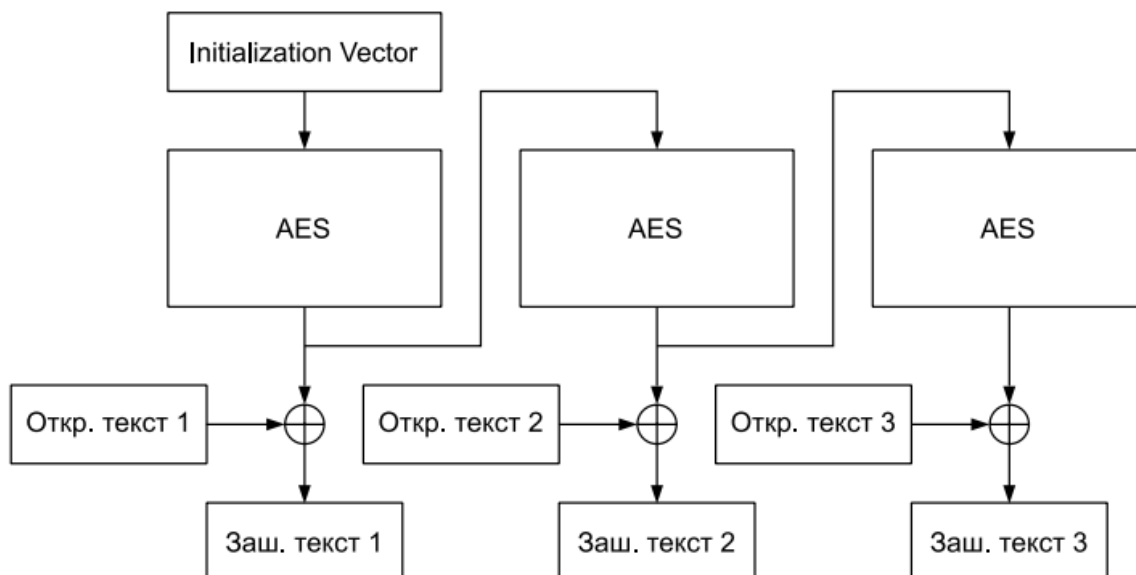


Рисунок 1.10 – Шифрование в режиме OFB

Шифрование и расшифрование в режиме OFB показаны на рисунке 1.10 и рисунке 1.11 соответственно

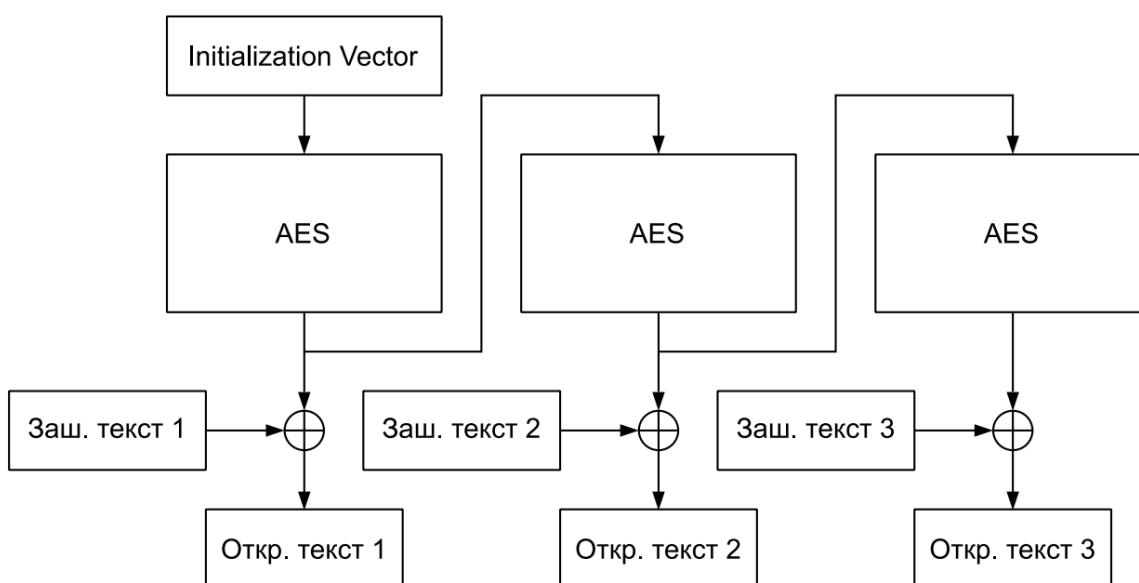


Рисунок 1.11 – Расшифрование в режиме OFB

Одновременное шифрование и расшифрование нескольких блоков невозможно, поскольку для применения шифрования к какому-либо блоку нужно зашифровать также и все предыдущие блоки.

## 2 Конструкторский раздел

### 2.1 Алгоритмы AES

На рисунке 2.1 представлена обобщенная схема шифрования в алгоритме AES.

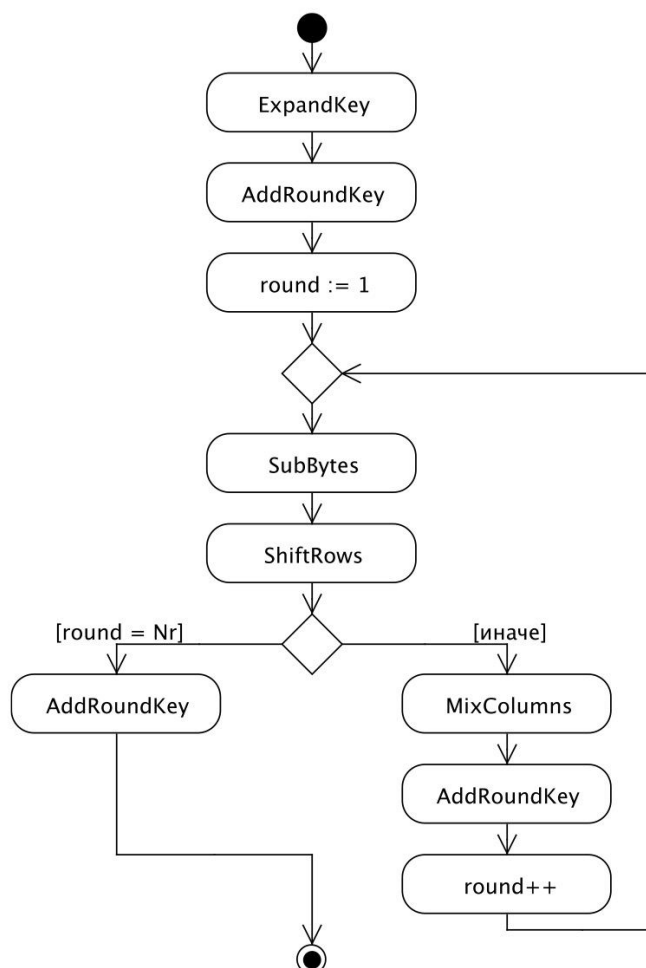


Рисунок 2.1 – Обобщенная схема шифрования в алгоритме AES

На рисунке 2.2 представлена обобщенная схема дешифрования в алгоритме AES.

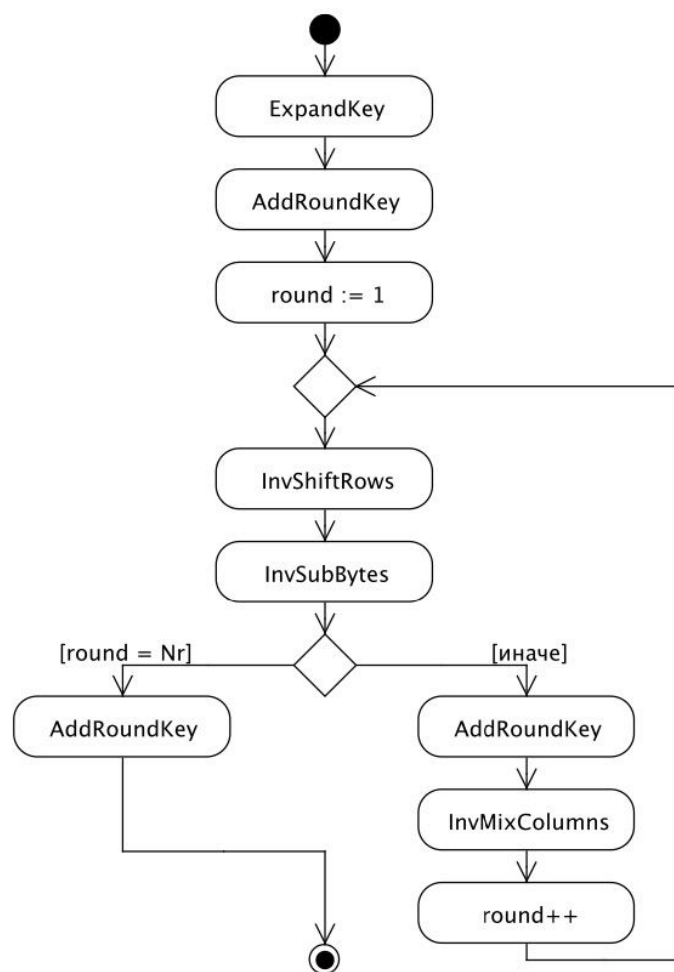


Рисунок 2.2 – Обобщенная схема дешифрования в алгоритме AES

## 3 Технологический раздел

### 3.1 Реализация алгоритмов

В листинге 3.1 представлен код алгоритма шифрования и дешифрования.

Листинг 3.1 – Исходный код шифрования

```
1 void aes_encrypt(const uint8_t *in, uint8_t *out, uint8_t *key) {
2     uint8_t state[WORD_SIZE * N_B];
3     block_to_state(in, state);
4
5     uint8_t expanded_key[WORD_SIZE * N_B * (N_R + 1)];
6     expand_key(key, expanded_key);
7     add_round_key(state, expanded_key, 0);
8
9     for (uint8_t r = 1; r < N_R; r++) {
10         sub_bytes(state);
11         shift_rows(state);
12         mix_columns(state);
13         add_round_key(state, expanded_key, r);
14     }
15
16     sub_bytes(state);
17     shift_rows(state);
18     add_round_key(state, expanded_key, N_R);
19
20     block_from_state(state, out);
21 }
```

В листинге 3.2 представлена реализация режима шифрования OBF.

Листинг 3.2 – Реализация режима шифрования OBF

```
1 void ofb(uint8_t *data, int num_blocks, uint8_t *iv, uint8_t *key) {
2     uint8_t iv_copy[BLOCK_SIZE + 1];
3
4     memcpy(iv_copy, iv, BLOCK_SIZE + 1);
5
6     for (int i = 0; i < num_blocks; i++) {
7         aes_encrypt(iv_copy, iv_copy, key);
8
9         for (int j = 0; j < BLOCK_SIZE; j++) {
10             data[BLOCK_SIZE * i + j] = data[BLOCK_SIZE * i + j] ^
11                 iv_copy[j];
12         }
13 }
```

## 3.2 Тестирование




**Входные данные:** имя входного файла, имя выходного файла, режим работы программы, ключ, начальный вектор.

**Выходные данные:** зашифрованный или расшифрованный файл в зависимости от режима работы программы.

Тестирование было проведено на файлах с типами: текстовый (txt), графический (bmp, png), архив (zip), несуществующий (ubs). Также, был проведен тест с повреждением зашифрованного файла.

В таблице 3.1 представлены тестовые данные.

Таблица 3.1 – Тестовые данные

Номер теста	Тип файла	Содержимое файла
1	txt	Наглая Пугачева
2	ubc	∅
3	zip	Файлы с тестов 1, 2, 4
4	png	
5	bmp	
6	bmp (corrupted) (in english)	



## ЗАКЛЮЧЕНИЕ

В ходе выполнения данной лабораторной работы поставленная цель была достигнута: было разработано программное обеспечение, позволяющее шифровать и дешифровать произвольный файл по алгоритму симметричного шифрования AES. Были выполнены все задачи:

- изучен алгоритм симметричного шифрования AES;
- реализован алгоритм симметричного шифрования AES;
- реализован алгоритм режим OFB.