МЕТОДЫ СИНТЕЗА И АНАЛИЗА СИММЕТРИЧНЫХ КРИПТОГРАФИЧЕСКИХ ПРЕОБРАЗОВАНИЙ

УДК 621.3.06

І.Д. ГОРБЕНКО, д-р техн. наук, Р.В. ОЛІЙНИКОВ, д-р техн. наук,
О.В. КАЗИМИРОВ, канд. техн. наук, В.І. РУЖЕНЦЕВ, канд. техн. наук,
О.О. КУЗНЄЦОВ, д-р техн. наук, Ю.І. ГОРБЕНКО, канд. техн. наук,
О.В. ДИРДА, канд. техн. наук, В.І. ДОЛГОВ, д-р техн. наук, А.І. ПУШКАРЬОВ,
Р.І. МОРДВІНОВ, канд. техн. наук, Д.С. КАЙДАЛОВ, В.М. КАЗИМИРОВА

СИМЕТРИЧНИЙ БЛОКОВИЙ ШИФР „КАЛИНА” – НОВИЙ НАЦІОНАЛЬНИЙ СТАНДАРТ УКРАЇНИ

Вступ

У якості основного стандарту блокового шифрування в Україні з 1990 р. використовувався ДСТУ ГОСТ 28147:2009 (ГОСТ 28147-89 [1]). Зараз це перетворення ще забезпечує практичну стійкість у режимах забезпечення конфіденційності, водночас, з точки зору швидкодії реалізації на програмних платформах загального призначення, суттєво поступається більш сучасним рішенням, таким як AES [2] , що призводить до ускладнення та подорожчання засобів КЗІ при однакових інших характеристиках. Крім того, для ГОСТ 28147-89 відомі теоретичні атаки, більш ефективні, ніж повний перебір ключів [3].

Враховуючи сучасні міжнародні тенденції та позитивний досвід у галузі розробки перспективних криптографічних перетворень, Державна служба спеціального зв’язку та захисту інформації України успішно провела національний відкритий конкурс симетричних блокових криптографічних алгоритмів [4]. За результатами конкурсу відзначений алгоритм „Калина”, на базі якого був розроблений національний стандарт ДСТУ 7624:2014 [5].

Крім блокового шифру ДСТУ 7624:2014 визначає режими роботи для забезпечення конфіденційності та цілісності, значення для перевірки, надає рекомендації щодо реалізації криптографічного перетворення і обмежень на обсяг інформації, яка захищається на одному ключі.

Для скорочення обсягу тексту національного стандарту була застосована математична нотація, що дозволяє отримати точний і компактний запис. Водночас, такий підхід може ускладнювати сприйняття сутності алгоритму для фахівців, що не мають фундаментальної криптологічної освіти. В статті наводиться розгорнутий альтернативний опис блокового шифру „Калина”, із позначеннями, традиційними для галузі комп’ютерних наук.

1. Термінологія та позначення

Відкритий текст	Блок даних розміром 128 біт, 256 біт, або 512 біт, що підлягає зашифруванню, а також блок даних такого ж розміру після розшифрування; (розміри відкритого тексту та шифртексту співпадають).
Шифртекст	Блок даних розміром 128 біт, 256 біт, або 512 біт після зашифрування, або це й же блок даних, що підлягає розшифруванню (розміри відкритого тексту та шифртексту співпадають).
Зашифрування	Перетворення відкритого тексту в шифртекст.
Розшифрування	Перетворення, обернене до зашифрування.
Шифрування	Зашифрування та/або розшифрування.

Ключ шифрування K	Блок даних розміром 128 біт, 256 біт, або 512 біт, що використовується в якості змінного секретного параметру в процедурі зашифрування або розшифрування.
Розмір блоку N_b	Довжина блоку відкритого тексту, виражена 64-бітними елементами (N_b може набувати значеннь 2, 4 або 8 відповідно для 128-бітного, 256-бітного або 512-бітного блоку); $N_b = \frac{l}{64}$, де l – розмір внутрішнього стану (біти) базового перетворення в ДСТУ 7624:2014.
Довжина ключа N_k	Довжина ключа шифрування, виражена 64-бітними елементами (N_k може набувати значення 2, 4 або 8 відповідно для 128-бітного, 256-бітного або 512-бітного ключа); $N_k = \frac{k}{64}$, де k – довжина ключа (біти) в ДСТУ 7624:2014.
Кількість циклів шифрування N_r	Число циклових перетворень при шифруванні, що визначається довжиною ключа та розміром блоку ($N_r = t$ в ДСТУ 7624:2014).
Процедура формування циклових ключів	Алгорим перетворення ключа шифрування, призначений для формування циклових ключів, що використовуються в циклових перетвореннях, попередньому та прикінцевому забілюванні (рандомізації).
Поточний стан S	Блок даних, що отримується на проміжних етапах виповнення процедури зашифрування або розшифрування (розмір поточного стану співпадає з розміром відкритого тексту).
Допоміжний ключ K_σ	Блок даних, що отримується з ключа шифрування та використовується для формування циклових ключів. Розмір проміжного ключа співпадає з розміром блоку.
Цикловий ключ K_i	Блок даних, отриманий з ключа шифрування у результаті виконання процедури формування циклових ключів (з використанням допоміжного ключа). Розмір циклового ключа співпадає з розміром блока відкритого тексту.

Далі використовуються наступні позначення:

<<<	циклічний побайтовий зсув вліво;
[+]	складання 64-бітних слів за модулем 2^{64};
$\oplus$	побітове складання двійкових векторів за модулем 2 (XOR);
0x	– префікс числа, що записане у шістнадцятковій системі числення;
B_i	– i -й байт вхідної послідовності;
$c(x)$	– многочлен над полем $GF(2^8)$;
$f(x)$	– многочлен над полем $GF(2)$;
$GF(2^8)$	– розширення простого поля степіня 8;

2. Загальні положення

Визначення алгоритму криптографічного перетворення містить опис процедури зашифрування, розшифрування, формування циклових ключів.

Алгоритм шифрування виконує обробку блоків даних під керуванням ключів:

- блок розміром 128 біт (ключ довжиною 128 і 256 біт);
- блок розміром 256 біт (ключ довжиною 256 і 512 біт);
- блок розміром 512 біт (ключ довжиною 512 біт).

2.1. Представлення вхідних та вихідних даних

До вхідних та вихідних даних алгоритму «Калина» належать відкритий текст та шифртекст відповідно. Ці параметри представляються у вигляді рядків заданої довжини $8 \times N_b$ байт ($64 \times N_b$ біт). Додатковим вхідним параметром є ключ, розмір якого дорівнює $8 \times N_k$ байт ($64 \times N_k$ біт).

Байтовий рядок довжиною $n = 8 \times N_b$ байт представляється у наступній формі:

$$B_0 B_1 B_2 \ldots B_{n-1}.$$

2.2. Поточний стан шифра

При виконанні зашифрування або розшифрування операції виконуються над двомірним масивом байт (поточним станом шифра).

Поточний стан можна представити у вигляді матриці розмірністю $8 \times N_b$ байт (вісім рядків довжиною N_b байт). Кожний байт в поточному стані має два індекси: номер рядка (r, $0 \le r < 8$) і номер стовпця (c, $0 \le c < N_b$). Байт адресується як $s_{r,c}$ або $s[r,c]$.

Крім того, у ряді операцій поточний стан шифру $S = (s_{i,j})$ розглядається як послідовність 64-бітних слів $S = (S_i)$. У цьому випадку кожний стовпець з номером i, $0 \le i < N_b$, розглядається як окреме слово, що має значення $S_i = s_{0,i} \cdot 2^{0 \cdot 8} + s_{1,i} \cdot 2^{1 \cdot 8} + \ldots + s_{7,i} \cdot 2^{7 \cdot 8}$. Відповідно, молодшим бітом 64-бітного блока є молодший біт байта рядка з найменшим номером, старшим бітом 64-бітного блока – старший біт байта із рядка з найбільшим номером.

2.3. Заповнення поточного стану

До початку зашифрування відкритий текст копіюється в поточний стан шифра. Після завершення процедури зашифрування шифртекст копіюється з поточного стану.

Довжина блока 128 біт.

Відкритий текст представлено байтовою послідовністю $in_0, in_1, \ldots in_{15}$. Отриманий шифртекст представлено як послідовність байт $out_0, out_1, \ldots out_{15}$.

Заповнення поточного стану шифра перед початком зашифрування та після його закінчення представлено на рис. 1.

Аналогічно, при розшифруванні шифртекст позначається байтовою послідовністю $in_0, in_1, \ldots in_{15}$; отриманий відкритий текст представлено послідовністю байт $out_0, out_1, \ldots out_{15}$; заповнення відповідає рис.1.

Вхідна послідовність			Поточний стан			Вихідна послідовність	
in_0	in_8		$s_{0,0}$	$s_{0,1}$		out_0	out_8
in_1	in_9		$s_{1,0}$	$s_{1,1}$		out_1	out_9
in_2	in_{10}		$s_{2,0}$	$s_{2,1}$		out_2	out_{10}
in_3	in_{11}	→	$s_{3,0}$	$s_{3,1}$	→	out_3	out_{11}
in_4	in_{12}		$s_{4,0}$	$s_{4,1}$		out_4	out_{12}
in_5	in_{13}		$s_{5,0}$	$s_{5,1}$		out_5	out_{13}
in_6	in_{14}		$s_{6,0}$	$s_{6,1}$		out_6	out_{14}
in_7	in_{15}		$s_{7,0}$	$s_{7,1}$		out_7	out_{15}

Рис. 1. Заповнення поточного стану для довжини блока 128 біт

Довжина блока 256 біт.

Відкритий текст представлено байтовим рядком $in_0, in_1, \ldots in_{31}$. Отриманий шифртекст представлено як послідовність байт $out_0, out_1, \ldots out_{31}$.

Заповнення поточного стану шифра перед початком зашифрування та після його закінчення представлено на рис. 2.

При розшифруванні шифртекст позначається байтовою послідовністю $in_0, in_1, \ldots in_{31}$; отриманий відкритий текст представлено послідовністю байт $out_0, out_1, \ldots out_{31}$; заповнення відповідає рис. 2.

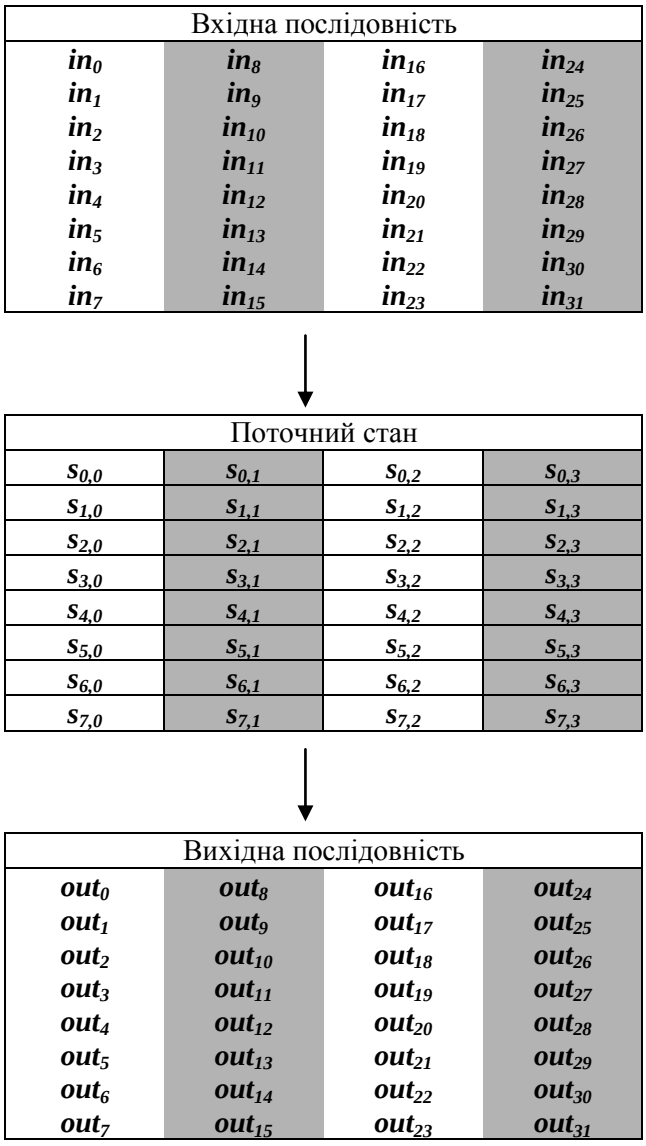


Рис. 2. Заповнення поточного стану для довжини блока 256 біт

Довжина блока 512 біт.

Відкритий текст представлено байтовою послідовністю $in_0, in_1, \ldots in_{63}$. Отриманий шифртекст представлено як послідовність байт $out_0, out_1, \ldots out_{63}$.

Заповнення поточного стану шифра перед початком зашифрування та після його закінчення представлено на рис. 3.

При розшифруванні шифртекст позначається байтовою послідовністю $in_0, in_1, \ldots in_{63}$; отриманий відкритий текст представлено послідовністю байт $out_0, out_1, \ldots out_{63}$; заповнення відповідає рис. 3.

Вхідна послідовність							
in_0	in_8	in_{16}	in_{24}	in_{32}	in_{40}	in_{48}	in_{56}
in_1	in_9	in_{17}	in_{25}	in_{33}	in_{41}	in_{49}	in_{57}
in_2	in_{10}	in_{18}	in_{26}	in_{34}	in_{42}	in_{50}	in_{58}
in_3	in_{11}	in_{19}	in_{27}	in_{35}	in_{43}	in_{51}	in_{59}
in_4	in_{12}	in_{20}	in_{28}	in_{36}	in_{44}	in_{52}	in_{60}
in_5	in_{13}	in_{21}	in_{29}	in_{37}	in_{45}	in_{53}	in_{61}
in_6	in_{14}	in_{22}	in_{30}	in_{38}	in_{46}	in_{54}	in_{62}
in_7	in_{15}	in_{23}	in_{31}	in_{39}	in_{47}	in_{55}	in_{63}

↓

Поточний стан							
$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$	$s_{0,4}$	$s_{0,5}$	$s_{0,6}$	$s_{0,7}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$	$s_{1,4}$	$s_{1,5}$	$s_{1,6}$	$s_{1,7}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$	$s_{2,4}$	$s_{2,5}$	$s_{2,6}$	$s_{2,7}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$	$s_{3,4}$	$s_{3,5}$	$s_{3,6}$	$s_{3,7}$
$s_{4,0}$	$s_{4,1}$	$s_{4,2}$	$s_{4,3}$	$s_{4,4}$	$s_{4,5}$	$s_{4,6}$	$s_{4,7}$
$s_{5,0}$	$s_{5,1}$	$s_{5,2}$	$s_{5,3}$	$s_{5,4}$	$s_{5,5}$	$s_{5,6}$	$s_{5,7}$
$s_{6,0}$	$s_{6,1}$	$s_{6,2}$	$s_{6,3}$	$s_{6,4}$	$s_{6,5}$	$s_{6,6}$	$s_{6,7}$
$s_{7,0}$	$s_{7,1}$	$s_{7,2}$	$s_{7,3}$	$s_{7,4}$	$s_{7,5}$	$s_{7,6}$	$s_{7,7}$

↓

Вихідна послідовність							
out_0	out_8	out_{16}	out_{24}	out_{32}	out_{40}	out_{48}	out_{56}
out_1	out_9	out_{17}	out_{25}	out_{33}	out_{41}	out_{49}	out_{57}
out_2	out_{10}	out_{18}	out_{26}	out_{34}	out_{42}	out_{50}	out_{58}
out_3	out_{11}	out_{19}	out_{27}	out_{35}	out_{43}	out_{51}	out_{59}
out_4	out_{12}	out_{20}	out_{28}	out_{36}	out_{44}	out_{52}	out_{60}
out_5	out_{13}	out_{21}	out_{29}	out_{37}	out_{45}	out_{53}	out_{61}
out_6	out_{14}	out_{22}	out_{30}	out_{38}	out_{46}	out_{54}	out_{62}
out_7	out_{15}	out_{23}	out_{31}	out_{39}	out_{47}	out_{55}	out_{63}

Рис. 3. Заповнення поточного стану для довжини блока 512 біт

2.4. Незвідний поліном шифра

Операція лінійного розсіювання (перемішування в стовпці) алгоритму «Калина» використовує поліноміальне представлення байт в полі GF(2^8), сформованому незвідним поліномом. Для алгоритму шифрування «Калина» у якості незвідного полінома використовується

$$f(x) = x^8 + x^4 + x^3 + x^2 + 1,$$

або {01}{1d} у шістнадцятковому представленні.

2.5. Розмір блока та довжина ключа шифрування

Алгоритм шифрування «Калина» виконує криптографічне перетворення блоків інформації розміром 128, 256 і 512 біт, використовуючи ключ шифрування довжиною 128, 256 і 512 біт. Довжина ключа співпадає з розміром блока або в два рази перебільшує його. Допустимі комбінації розміру блока та довжини ключа шифрування наведено у табл. 1.

Таблиця 1

Комбінації розміру блока та довжини ключа шифрування

Розмір блоку, біт	Довжина ключа, біт
128 (N_b = 2)	128 (N_k = 2), 256 (N_k = 4)
256 (N_b = 4)	256 (N_k = 4), 512 (N_k = 8)
512 (N_b = 8)	512 (N_k = 8)

2.6. Кількість циклів шифрування

Алгоритм шифрування є процедурою, що складається з попереднього і прикінцевого забілювання та ітеративного циклового перетворення. На вхід кожного циклового перетворення подається поточний стан, а також необхідна кількість ключової інформації (цикловий ключ). Відкритий текст копіюється в поточний стан перед початком зашифрування, а після його завершення в поточному стані знаходиться шифртекст. Кількість циклів шифрування N_r залежить від довжини ключа; її наведено у табл. 2. На початку та у кінці процедури зашифрування/розшифрування виконуються додаткові операції забілювання.

Таблиця 2

Кількість циклів шифрування алгоритму «Калина» N_r

Довжина ключа (біт) / Розмір блока (біт)	128 (N_k = 2)	256 (N_k = 4)	512 (N_k = 8)
128 (N_b = 2)	10	14	–
256 (N_b = 4)	–	14	18
512 (N_b = 8)	–	–	18

3. Зашифрування у режимі простої заміни

На вхід процедури подається відкритий текст та циклові ключі. На початку зашифрування відкритий текст представляється у вигляді блока даних, що описує поточний стан
шифра (див. п. 2.2). Після закінчення зашифрування отриманий шифртекст формується у вигляді байтової послідовності (див. п. 2.2). Псевдокод процедури зашифрування алгоритму «Калина», що відповідає базовому перетворенню $T_{l,k}^{(K)}$ в ДСТУ 7624:2014, наведено на рис. 4.

```
void Kalyna_Cipher(byte in[8 * Nb], byte out[8 * Nb], byte roundkey[Nr + 1][8 *
Nb]) {
byte state[ 8, Nb ] = in

Add64RoundKey( state, roundkey[ 0 ] )

for(round = 1 to Nr-1 step 1) {
            Kalyna_S_boxes( state )
            ShiftRows( state )
            MixColumns( state )
            XORRoundKey( state, roundkey[ round ] )
}

Kalyna_S_boxes( state )
ShiftRows( state )
MixColumns( state )

Add64RoundKey( state, roundkey[ Nr ] )
out = state
}
```

Рис. 4. Псевдокод процедури зашифрування алгоритму «Калина»

При здійсненні перетворення XORRoundKey (функція $\kappa_l^{(K_v)}$ в ДСТУ 7624:2014) виконується побітове складання за модулем 2 (XOR) поточного стану та циклового ключа. Після виконання операції результат записується на місце першого аргументу.

Представлення циклового ключа як матриці віповідного розміру є аналогічним представленню відкритого тексту у вигляді поточного стану шифра (див. п. 2.2).

Для поточного стану $A=(a_{i,j})$ та циклового ключа $k=(k_{i,j})$ результат перетворення $B=(b_{i,j})=A\oplus k$ обчислюється за формулою $b_{i,j}=a_{i,j}\oplus k_{i,j}$, $0\le i<8$, $0\le j<N_b$.

Приклад здійснення перетворення для розміру блока 256 біт представлено на рис. 5.

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$\oplus$	$k_{0,0}$	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$=$	$b_{0,0}$	$b_{0,1}$	$b_{0,2}$	$b_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$		$k_{1,0}$	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$		$b_{1,0}$	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$		$k_{2,0}$	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$		$b_{2,0}$	$b_{2,1}$	$b_{2,2}$	$b_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$		$k_{3,0}$	$k_{3,1}$	$k_{3,2}$	$k_{3,3}$		$b_{3,0}$	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$
$a_{4,0}$	$a_{4,1}$	$a_{4,2}$	$a_{4,3}$		$k_{4,0}$	$k_{4,1}$	$k_{4,2}$	$k_{4,3}$		$b_{4,0}$	$b_{4,1}$	$b_{4,2}$	$b_{4,3}$
$a_{5,0}$	$a_{5,1}$	$a_{5,2}$	$a_{5,3}$		$k_{5,0}$	$k_{5,1}$	$k_{5,2}$	$k_{5,3}$		$b_{5,0}$	$b_{5,1}$	$b_{5,2}$	$b_{5,3}$
$a_{6,0}$	$a_{6,1}$	$a_{6,2}$	$a_{6,3}$		$k_{6,0}$	$k_{6,1}$	$k_{6,2}$	$k_{6,3}$		$b_{6,0}$	$b_{6,1}$	$b_{6,2}$	$b_{6,3}$
$a_{7,0}$	$a_{7,1}$	$a_{7,2}$	$a_{7,3}$		$k_{7,0}$	$k_{7,1}$	$k_{7,2}$	$k_{7,3}$		$b_{7,0}$	$b_{7,1}$	$b_{7,2}$	$b_{7,3}$

Рис. 5. Перетворення XORRoundKey для розміру блока 256 біт

При здійсненні перетворення Add64RoundKey (функція $\eta_l^{(K_v)}$ в ДСТУ 7624:2014) виконується складання за модулем 2^{64} 64-бітних слів поточного стану та циклового ключа. Після виконання операції результат записується на місце першого аргументу.

Представлення циклового ключа як матриці віповідного розміру є аналогічним представленню відкритого тексту у вигляді поточного стану шифра (див. п. 2.2). При розбитті
поточного стану та циклового ключа на 64-бітні блоки молодшим бітом 64-бітного блоку буде молодший біт байту рядка з найменшим номером, старшим бітом 64-бітного блоку – старший біт байта рядка з найбільшим номером (формат little endian). Аналогічним чином проходить розбиття циклового ключа на 64-бітні блоки, після чого виконується складання відповідних блоків за модулем 2^{64}.

Для поточного стану $A=(a_i)$, де $a_i=a_{0,i}\cdot 2^{0\cdot 8}+a_{1,i}\cdot 2^{1\cdot 8}+\ldots+a_{7,i}\cdot 2^{7\cdot 8}$ та $0\le i<N_b$, циклового ключа $k=(k_i)$ результат перетворення $B=(b_i)$ обчислюється за формулою $b_i=a_i+k_i\pmod{2^{64}}$.

Схему розбиття на 64-бітні блоки 128-бітного стану та циклового ключа такого ж розміру з наступним перетворенням Add64RoundKey наведено на рис. 6.

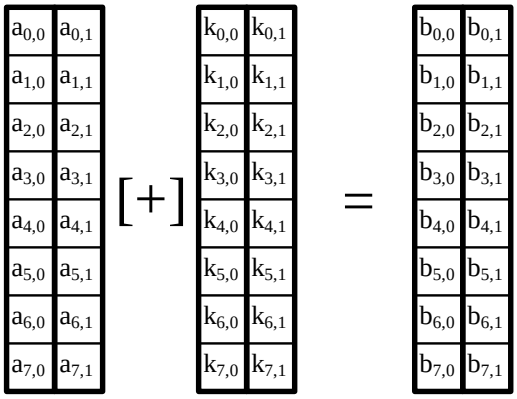


Рис. 6. Перетворення Add64RoundKey для розміру блока128 біт

Перетворення Add64RoundKey для станів розміром 256 і 512 біт буде відрізнятися лише кількістю 64-бітних блоків, на які розбивається стан та цикловий ключ (приклад для 256-бітного стану наведено на рис. 7).

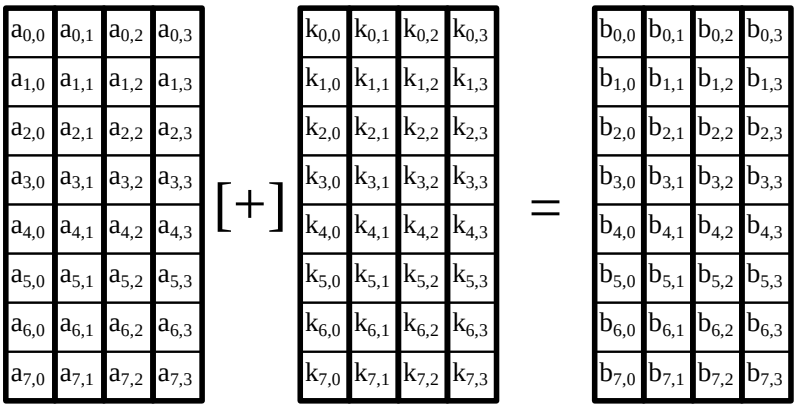


Рис. 7. Перетворення Add64RoundKey для розміру блока 256 біт

Перетворення Kalyna_S_boxes (шар нелінійного бієктивного відображення π_l' в ДСТУ 7624:2014) виконує заміну кожного байта поточного стану у відповідності з заданою таблицею підстановки. Використовуються чотири різні підстановки «байт-в-байт», причому для байтів одного рядка поточного стану шифра використовується одна і та ж підстановка:

для байт 0-го рядка (елементи $\boldsymbol{s_{0,i}}$) – підстановка *S0*;
для байт 1-го рядка (елементи $\boldsymbol{s_{1,i}}$) – підстановка *S1*;
для байт 2-го рядка (елементи $\boldsymbol{s_{2,i}}$) – підстановка *S2*;
для байт 3-го рядка (елементи $\boldsymbol{s_{3,i}}$) – підстановка *S3*;
для байт 4-го рядка (елементи $\boldsymbol{s_{4,i}}$) – підстановка *S0*;
для байт 5-го рядка (елементи $\boldsymbol{s_{5,i}}$) – підстановка *S1*;
для байт 6-го рядка (елементи $\boldsymbol{s_{6,i}}$) – підстановка *S2*;
для байт 7-го рядка (елементи $\boldsymbol{s_{7,i}}$) – підстановка *S3*.

Таблиці підстановки алгоритму «Калина» наведено в додатку до статті (підстановки π_0 - π_3 в ДСТУ 7624:2014). На рис. 8 – приклад заміни байта при виконанні перетворення.

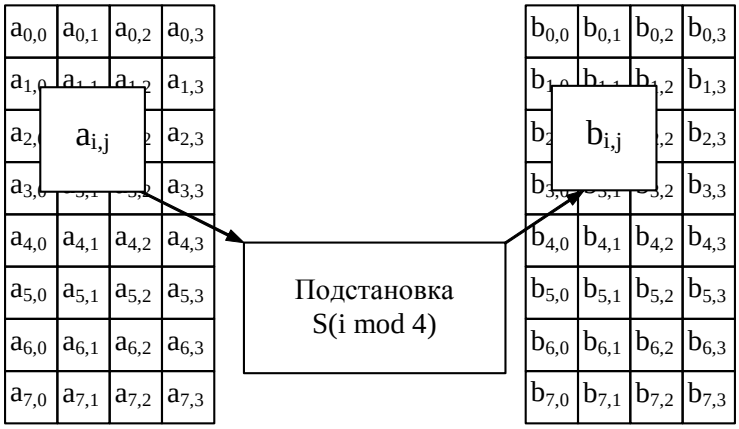


Рис. 8. Приклад підстановки байта для розміра блоку 256 біт

Заміна одного байта полягає у виборі з таблиці підстановки нового значення за адресою із зсувом, що задає поточне значення байта. Нове вибране значення і є результатом здійснення підстановки для одного байта.

Приклад підстановки для таблиці S0 байта з шістнадцятковим значенням 5A наведено у табл. 3.

Таблиця 3

Приклад підстановки для таблиці S0

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	A8	43	5F	06	6B	75	6C	59	71	DF	87	95	17	F0	D8	09
1	6D	F3	1D	CB	C9	4D	2C	AF	79	E0	97	FD	6F	4B	45	39
2	3E	DD	A3	4F	B4	B6	9A	0E	1F	BF	15	E1	49	D2	93	C6
3	92	72	9E	61	D1	63	FA	EE	F4	19	D5	AD	58	A4	BB	A1
4	DC	F2	83	37	42	E4	7A	32	9C	CC	AB	4A	8F	6E	04	27
5	2E	E7	E2	5A	96	16	23	2B	C2	65	66	0F	BC	A9	47	41
6	34	48	FC	B7	6A	88	A5	53	86	F9	5B	DB	38	7B	C3	1E
7	22	33	24	28	36	C7	B2	3B	8E	77	BA	F5	14	9F	08	55
8	9B	4C	FE	60	5C	DA	18	46	CD	7D	21	B0	3F	1B	89	FF
9	EB	84	69	3A	9D	D7	D3	70	67	40	B5	DE	5D	30	91	B1
A	78	11	01	E5	00	68	98	A0	C5	02	A6	74	2D	0B	A2	76
B	B3	BE	CE	BD	AE	E9	8A	31	1C	EC	F1	99	94	AA	F6	26
C	2F	EF	E8	8C	35	03	D4	7F	FB	05	C1	5E	90	20	3D	82
D	F7	EA	0A	0D	7E	F8	50	1A	C4	07	57	B8	3C	62	E3	C8
E	AC	52	64	10	D0	D9	13	0C	12	29	51	B9	CF	D6	73	8D
F	81	54	C0	ED	4E	44	A7	2A	85	25	E6	CA	7C	8B	56	80

Старші чотири біти визначають рядок, молодші чотири біти – стовпець. Результат підстановки для значення 5A – це число в шістнадцятковому представленні 66, що знаходиться в таблиці на перетині 6-го рядка (з індексом 5) та 11-го стовпця (з індексом A).

Для здійснення перетворення може використовуватися інший набір підстановок (від 1 до 8 таблиць), відмінний від наведеного у додатку. У цьому випадку набір підстановок має постачатися в установленому порядку і може бути додатковим секретним параметром шифра, як і ключ шифрування.

Під час перетворення ShiftRows (перестановка елементів τ_l в ДСТУ 7624:2014)

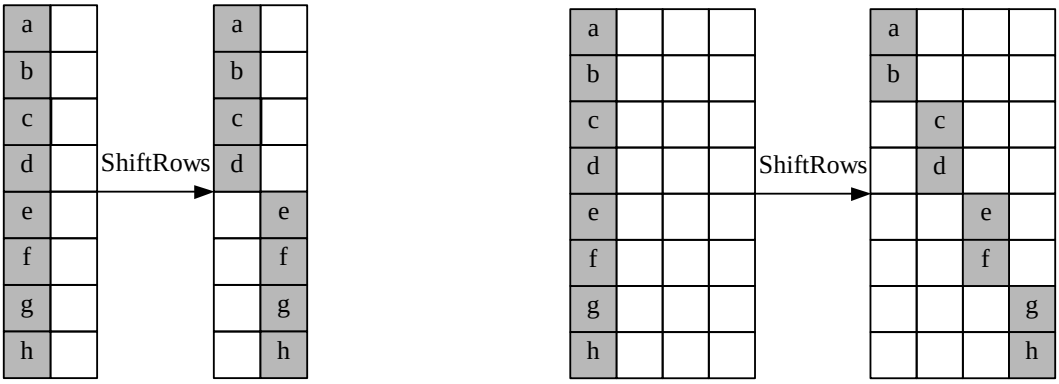
здійснюється рівномірне розподілення байт кожного 64-бітного стовпця серед інших стовпців. Це досягається шляхом циклічного зсуву рядків стану вправо на різну кількість байт. Значення зсувів залежать від розміру блока шифрування (табл. 4).

Таблиця 4

Значення циклічних зсувів рядків для різних розмірів блока

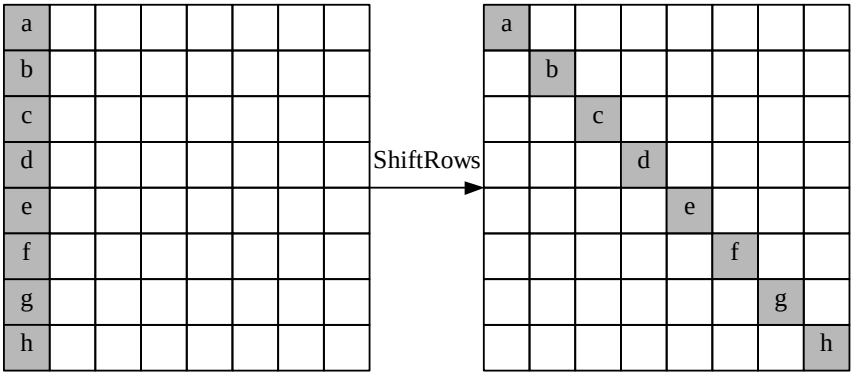
Номер рядка	Значення зсуву, байти		
	Довжина блоку128 біт	Довжина блоку 256 біт	Довжина блоку 512 біт
0	0	0	0
1	0	0	1
2	0	1	2
3	0	1	3
4	1	2	4
5	1	2	5
6	1	3	6
7	1	3	7

Рис. 9 пояснює порядок виконання перетворення ShiftRows для різних розмірів блоку.



а – 128-бітний блок

б – 256-бітний блок



в – 512-бітний блок

Рис. 9. Порядок розподілення байт першого стовпця при здійсненні перетворення ShiftRows

Під час здійснення перетворення MixColumns (лінійне перетворення ψ_l в ДСТУ 7624:2014) виконується послідовна обробка всіх стовпців поточного стану. Кожний 8-байтний стовпець розглядається як поліном над полем GF(2^8), що складається з суми восьми одночленів (кожний байт представляється у вигляді елемента поля GF(2^8) і є коефіцієнтом перед змінною степеня, що дорівнює індексу байта у стовпці). При перетворенні виконується множення цього полінома за модулем x^8+1 на фіксований поліном C(x), де

$$C(x) = \{01\}x^7+\{05\}x^6+\{01\}x^5+\{08\}x^4+\{06\}x^3+\{07\}x^2+\{04\}x+\{01\}.$$

Ця операція еквівалентна матричному множенню над $GF(2^8)$ початкового 8-байтового вектора *a* на фіксовану матрицю, результат зберігається у 8-байтний вектор *b* (див. рис. 10).

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} = \begin{bmatrix} 01 & 01 & 05 & 01 & 08 & 06 & 07 & 04 \\ 04 & 01 & 01 & 05 & 01 & 08 & 06 & 07 \\ 07 & 04 & 01 & 01 & 05 & 01 & 08 & 06 \\ 06 & 07 & 04 & 01 & 01 & 05 & 01 & 08 \\ 08 & 06 & 07 & 04 & 01 & 01 & 05 & 01 \\ 01 & 08 & 06 & 07 & 04 & 01 & 01 & 05 \\ 05 & 01 & 08 & 06 & 07 & 04 & 01 & 01 \\ 01 & 05 & 01 & 08 & 06 & 07 & 04 & 01 \end{bmatrix} \times \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{bmatrix}$$

Рис. 10. Матричне представлення перемішування у стовпці

Порядок обчислення елементів результуючого вектора *b* пояснюється на рис. 11, при цьому всі операції множення на байт виконуються у полі $GF(2^8)$, заданному в п.2.4.

На рис. 12 пояснюється порядок виконання перетворення MixColumns для поточного стану шифра.

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} = \begin{bmatrix} 01\cdot a_0 \oplus 01\cdot a_1 \oplus 05\cdot a_2 \oplus 01\cdot a_3 \oplus 08\cdot a_4 \oplus 06\cdot a_5 \oplus 07\cdot a_6 \oplus 04\cdot a_7 \\ 04\cdot a_0 \oplus 01\cdot a_1 \oplus 01\cdot a_2 \oplus 05\cdot a_3 \oplus 01\cdot a_4 \oplus 08\cdot a_5 \oplus 06\cdot a_6 \oplus 07\cdot a_7 \\ 07\cdot a_0 \oplus 04\cdot a_1 \oplus 01\cdot a_2 \oplus 01\cdot a_3 \oplus 05\cdot a_4 \oplus 01\cdot a_5 \oplus 08\cdot a_6 \oplus 06\cdot a_7 \\ 06\cdot a_0 \oplus 07\cdot a_1 \oplus 04\cdot a_2 \oplus 01\cdot a_3 \oplus 01\cdot a_4 \oplus 05\cdot a_5 \oplus 01\cdot a_6 \oplus 08\cdot a_7 \\ 08\cdot a_0 \oplus 06\cdot a_1 \oplus 07\cdot a_2 \oplus 04\cdot a_3 \oplus 01\cdot a_4 \oplus 01\cdot a_5 \oplus 05\cdot a_6 \oplus 01\cdot a_7 \\ 01\cdot a_0 \oplus 08\cdot a_1 \oplus 06\cdot a_2 \oplus 07\cdot a_3 \oplus 04\cdot a_4 \oplus 01\cdot a_5 \oplus 01\cdot a_6 \oplus 05\cdot a_7 \\ 05\cdot a_0 \oplus 01\cdot a_1 \oplus 08\cdot a_2 \oplus 06\cdot a_3 \oplus 07\cdot a_4 \oplus 04\cdot a_5 \oplus 01\cdot a_6 \oplus 01\cdot a_7 \\ 01\cdot a_0 \oplus 05\cdot a_1 \oplus 01\cdot a_2 \oplus 08\cdot a_3 \oplus 06\cdot a_4 \oplus 07\cdot a_5 \oplus 04\cdot a_6 \oplus 01\cdot a_7 \end{bmatrix}$$

Рис. 11. Порядок обчислення байт при виконанні перемішування в стовпці

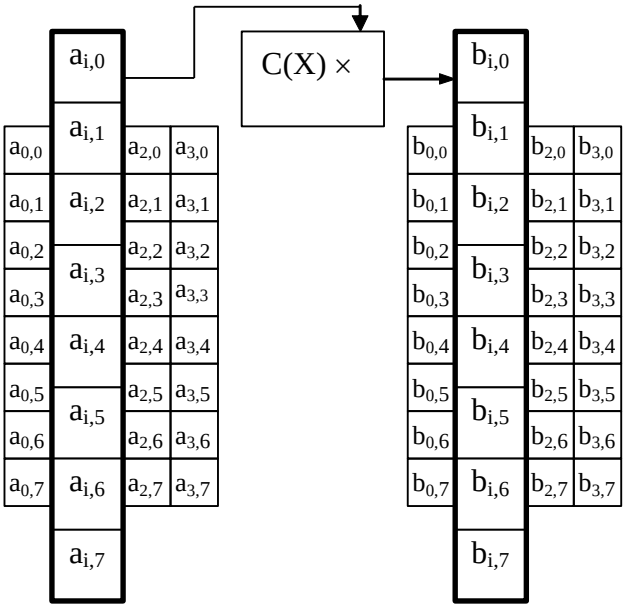


Рис. 12. Порядок виконання перетворення MixColumns
для поточного стану шифра з розміром блоку 256 біт

4. Розшифрування у режимі простої заміни

Процедура розшифрування є зворотною зашифруванню. На вхід подається шифртекст та циклові ключі. Після закінчення розшифрування отриманий відкритий текст формується у вигляді байтового рядка (див. п. 2.2). Псевдокод процедури розшифрування алгоритму «Калина», що відповідає базовому перетворенню $U_{l,k}^{(K)}$ в ДСТУ 7624:2014, наведено на рис.13.

Операція XORRoundKey є зворотною до себе (подвійне застосування дає початкове значення), відповідно, якщо до стану $B = A \oplus k$ (див. п. 3.2) додати цикловий ключ k, то буде отримано початковий стан A.

При зашифруванні та розшифруванні використовується одна й та сама операція XORRoundKey.

Зворотне для Add64RoundKey перетворення Sub64RoundKey (функція ${}_{-1}\eta_l^{(K_v)}$ в ДСТУ 7624:2014) полягає в аналогічному розбитті стану та циклового ключа на 64-бітні блоки та відніманні за модулем 2^{64} від блоків стану $B = (b_i)$ відповідних блоків циклового ключа $k = (k_i)$ для отримання стану (результату) $A = (a_i)$: $a_i = b_i - k_i \pmod{2^{64}}$, $0 \le i < N_b$.

```
void Kalyna_InvCipher(byte in[8 * Nb], byte out[8 * Nb], byte roundkey[Nr + 1][8
* Nb]) {
      byte state[ 8, Nb ] = in

      Sub64RoundKey(state, roundkey[  Nr  ]  )

      InvMixColumns( state )
      InvShiftRows( state )
      Kalyna_InvS_boxes( state )

      for(round = Nr - 1 to 1 downstep 1) {
            XORRoundKey(state, roundkey[ round ] )
            InvMixColumns( state )
            InvShiftRows( state )
            Kalyna_InvS_boxes( state )

      }

      Sub64RoundKey(state, roundkey[ 0 ] )
      out = state
}
```

Рис. 13. Псевдокод процедури розшифрування алгоритму «Калина»

Після виконання операції результат записується на місце першого параметра. Правила переходу від байт поточного стану до 64-бітних блоків та навпаки повністю ідентичні з перетвореннями, що виконуються під час Add64RoundKey.

Зворотне для Kalyna_S_boxes перетворення Kalyna_InvS_boxes (шар оберненого нелінійного бієктивного відображення ${}_{-1}\pi_l'$ в ДСТУ 7624:2014) полягає у заміні кожного байта поточного стану шифра відповідно до таблиць зворотних підстановок (підстановки ${}_{-1}\pi_0$ - ${}_{-1}\pi_3$ в ДСТУ 7624:2014). Таким чином, зворотна процедура відрізняється від прямої

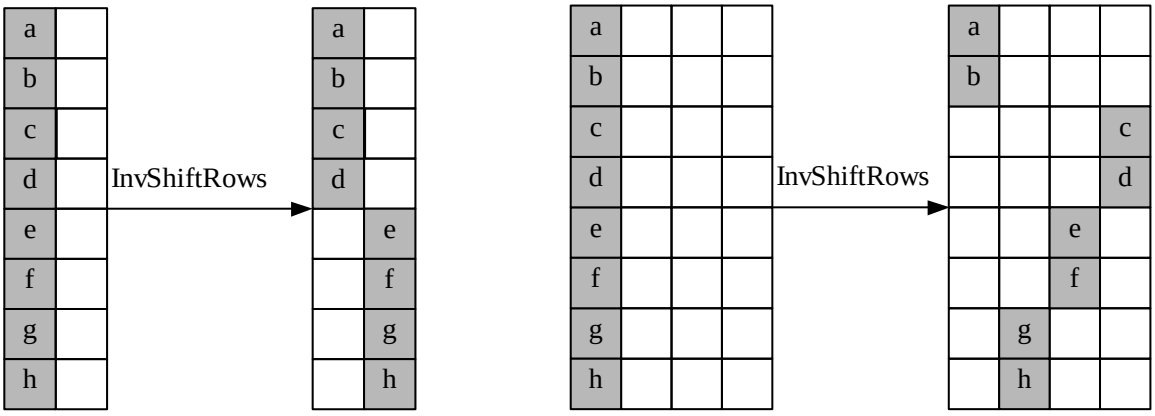
лише таблицями підстановок, а порядок виконання зворотного перетворення для кожного окремо взятого байта є повністю ідентичним порядку виконання прямого перетворення.

Якщо при зашифруванні використовувався набір підстановок, що відрізняється від наведеного у додатку, то при розшифруванні застосовується відповідний набір таблиць зворотної підстановки.

Зворотне для ShiftRows перетворення InvShiftRows (обернена перестановка елементів $_{-1}\tau_l$ в ДСТУ 7624:2014) полягає у виконанні циклічних зсувів рядків стану на ту ж саму кількість байт (табл. 4), але вліво. Рис. 14 пояснює порядок виконання перетворення InvShiftRows.

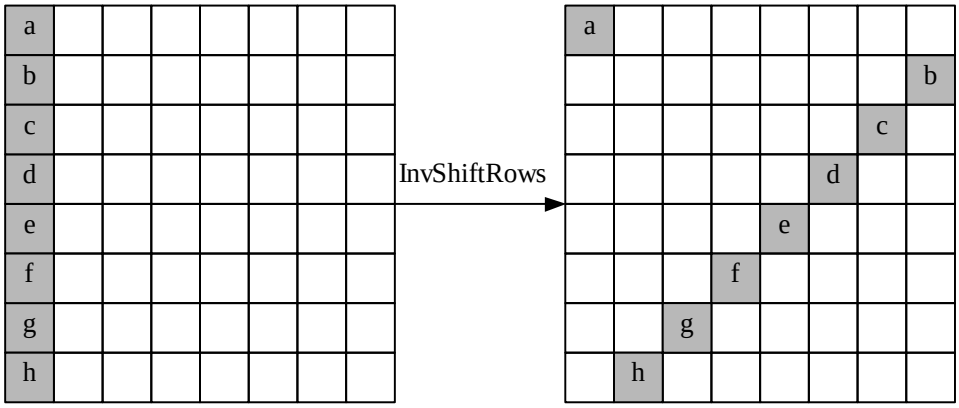
Зворотне для MixColumns перетворення InvMixColumns (обернене лінійне перетворення $_{-1}\psi_l$ в ДСТУ 7624:2014) полягає в перемноженні кожного стовпця, що представлений у вигляді полінома над полем GF(2^8), на зворотний для C(x) поліном D(x):

$$D(x) = \{95\}x^7+\{76\}x^6+\{A8\}x^5+\{2F\}x^4+\{49\}x^3+\{D7\}x^2+\{CA\}x+\{AD\}.$$



а – 128-бітний блок

б – 256-бітний блок



в – 512-бітний блок

Рис. 14. Порядок розподілення байт першого стовпця
при виконанні перетворення InvShiftRows

Операція еквівалентна матричному множенню над GF(2^8) початкового 8-байтового вектора *a* на фіксовану матрицю. Результат заноситься до 8-байтного вектору *b* (див. рис. 15).

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} = \begin{bmatrix} \text{AD} & 95 & 76 & \text{A8} & \text{2F} & 49 & \text{D7} & \text{CA} \\ \text{CA} & \text{AD} & 95 & 76 & \text{A8} & \text{2F} & 49 & \text{D7} \\ \text{D7} & \text{CA} & \text{AD} & 95 & 76 & \text{A8} & \text{2F} & 49 \\ 49 & \text{D7} & \text{CA} & \text{AD} & 95 & 76 & \text{A8} & \text{2F} \\ \text{2F} & 49 & \text{D7} & \text{CA} & \text{AD} & 95 & 76 & \text{A8} \\ \text{A8} & \text{2F} & 49 & \text{D7} & \text{CA} & \text{AD} & 95 & 76 \\ 76 & \text{A8} & \text{2F} & 49 & \text{D7} & \text{CA} & \text{AD} & 95 \\ 95 & 76 & \text{A8} & \text{2F} & 49 & \text{D7} & \text{CA} & \text{AD} \end{bmatrix} \times \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{bmatrix}$$

Рис. 15. Матричне представлення зворотного перемішування у стовпці

Порядок обчислення елементів результуючого вектора *b* пояснює рис. 16.

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} = \begin{bmatrix} \text{AD} \cdot a_0 \oplus 95 \cdot a_1 \oplus 76 \cdot a_2 \oplus \text{A8} \cdot a_3 \oplus \text{2F} \cdot a_4 \oplus 49 \cdot a_5 \oplus \text{D7} \cdot a_6 \oplus \text{CA} \cdot a_7 \\ \text{CA} \cdot a_0 \oplus \text{AD} \cdot a_1 \oplus 95 \cdot a_2 \oplus 76 \cdot a_3 \oplus \text{A8} \cdot a_4 \oplus \text{2F} \cdot a_5 \oplus 49 \cdot a_6 \oplus \text{D7} \cdot a_7 \\ \text{D7} \cdot a_0 \oplus \text{CA} \cdot a_1 \oplus \text{AD} \cdot a_2 \oplus 95 \cdot a_3 \oplus 76 \cdot a_4 \oplus \text{A8} \cdot a_5 \oplus \text{2F} \cdot a_6 \oplus 49 \cdot a_7 \\ 49 \cdot a_0 \oplus \text{D7} \cdot a_1 \oplus \text{CA} \cdot a_2 \oplus \text{AD} \cdot a_3 \oplus 95 \cdot a_4 \oplus 76 \cdot a_5 \oplus \text{A8} \cdot a_6 \oplus \text{2F} \cdot a_7 \\ \text{2F} \cdot a_0 \oplus 49 \cdot a_1 \oplus \text{D7} \cdot a_2 \oplus \text{CA} \cdot a_3 \oplus \text{AD} \cdot a_4 \oplus 95 \cdot a_5 \oplus 76 \cdot a_6 \oplus \text{A8} \cdot a_7 \\ \text{A8} \cdot a_0 \oplus \text{2F} \cdot a_1 \oplus 49 \cdot a_2 \oplus \text{D7} \cdot a_3 \oplus \text{CA} \cdot a_4 \oplus \text{AD} \cdot a_5 \oplus 95 \cdot a_6 \oplus 76 \cdot a_7 \\ 76 \cdot a_0 \oplus \text{A8} \cdot a_1 \oplus \text{2F} \cdot a_2 \oplus 49 \cdot a_3 \oplus \text{D7} \cdot a_4 \oplus \text{CA} \cdot a_5 \oplus \text{AD} \cdot a_6 \oplus 95 \cdot a_7 \\ 95 \cdot a_0 \oplus 76 \cdot a_1 \oplus \text{A8} \cdot a_2 \oplus \text{2F} \cdot a_3 \oplus 49 \cdot a_4 \oplus \text{D7} \cdot a_5 \oplus \text{CA} \cdot a_6 \oplus \text{AD} \cdot a_7 \end{bmatrix}$$

Рис. 16. Порядок обчислення байт за зворотного перемішування у стовпці

5. Формування циклових ключів

Для отримання циклових ключів застосовується процедура формування, що базується на цикловому перетворенні (для поточного розміру блоку). При шифруванні використовується $N_r + 1$ циклових ключів k_i ($i = 0,1,..., N_r$), кожний довжиною 64 $\times N_b$ біт (розмір циклового ключа співпадає з розміром відкритого/шифртексту і поточного стану шифра).

Алгоритм розгортання ключа працює за схемою, що містить три етапи:

1) з ключа шифрування K формується проміжний ключ K_σ з довжиною, що дорівнює розміру блока (64 $\times N_b$ біт) з використанням трьох циклів шифрування;

2) на основі ключа шифрування K та проміжного ключа K_σ формуються циклові ключі k_{2i} (з парними індексами) довжиною, що дорівнює розміру блока (*64* $\times N_b$ біт), з використанням двох циклів шифрування для кожного циклового ключа;

3) з циклових ключів k_{2i} з парними індексами формуються циклові ключі k_{2i+1} (з непарними індексами) шляхом циклічного зсуву попереднього ключа з парним індексом вліво на $2 \cdot N_b + 3$ байт.

Циклові ключі можуть формуватися незалежно один від одного з однаковою обчислювальною складністю як для зашифрування, так і для розшифрування.

Проміжний ключ K_σ має довжину 64 $\times N_b$ біт (співпадає за розміром з довжиною блока відкритого тексту) та формується на основі ключа шифрування K довжиною 64 $\times N_k$ біт з використанням операцій циклового перетворення.

Для формування K_σ використовуються три цикли шифрування. На вхід перетворення подається значення, що визначається поточним станом блока і довжиною ключа, а в якості циклових ключів використовується ключ шифрування (або його молодша та старша половини, якщо ключ є довшим за розмір вхідного блоку). Значення, отримане в результаті перетворення, є проміжним ключем K_σ. Псевдокод, що описує перетворення ($\Theta^{(K)}$ в ДСТУ 7624:2014), наведено на рис. 17.

Розмір значення для початкового заповнення змінної state дорівнює довжині блоку. У байт стану state[0, 0] заноситься арифметична сума ($N_b + N_k + 1$), в інші байти стану – 0.

При формуванні циклових ключів з парними індексами ключ шифрування K подається на вхід двох циклів шифрування як відкритий текст (один або два блоки, в залежності від довжини). У якості циклового ключа використовується результат обробки проміжного ключа K_σ функцією Add64RoundKey, де другим аргументом є значення, що залежить від індекса циклового ключа, що формується. Псевдокод, що описує формування циклових ключів з парними індексами ($\Xi^{(K,K_\sigma,i)}$ в ДСТУ 7624:2014), наведено на рис. 18.

```
void Kalyna_KeyExpansion_Ksigma (byte key[8 * Nk], byte Ksigma[8 * Nb])
{
      byte state[ 8, Nb ] = ( Nb + Nk + 1 ), 0, 0, ... , 0

byte k0[ 8 * Nb ], k1[ 8 * Nb ]

      if ( Nb == Nk )
      {
            k0 = key
            k1 = key
}
else
{
            k0 = key[ 0 .. 8 * Nb – 1 ]
            k1 = key[ 8 * Nb .. 8 * Nk – 1 ]
}

Add64RoundKey( state, k0 )

      Kalyna_S_boxes( state )
      ShiftRows( state )
      MixColumns( state )

      XORRoundKey( state, k1 )

      Kalyna_S_boxes( state )
      ShiftRows( state )
      MixColumns( state )

      Add64RoundKey( state, k0 )

      Kalyna_S_boxes( state )
      ShiftRows( state )
      MixColumns( state )

      Ksigma = state
}
```

Рис. 17. Псевдокод процедури обчислення проміжного ключа K_σ

Розмір змінної tmp_modification_value ($\vartheta = \mu_l^{(0x00010001...0001)}$ в ДСТУ 7624:2014) дорівнює довжині блоку. Константа формується повторенням байт 0x01, 0x00 (в шістнадцятковому представленні) до заповнення стану.

Операція ShiftLeft (<< в ДСТУ 7624:2014, функція $\varphi_i = \eta_l^{(\vartheta << (i/2))}$) оброблює стан шифра як послідовність 64-бітних слів. Кожне слово стану $(w_0, w_1, ..., w_{N_b-1})$, незалежно від інших, логічно зсувається вліво на 1 біт, тобто $w_i = 2 \cdot w_i \pmod{2^{64}}$.

Операція Rotate (>>> в ДСТУ 7624:2014) оброблює стан як послідовність 64-бітних слів $(w_0, w_1, ..., w_{N_k-1})$, виконуючи циклічний зсув та повертаючи стан $(w_1, ..., w_{N_k-1}, w_0)$.

Кожний із циклових ключів з парним індексом, що отриманий на попередньому етапі процедури формування, представляється у вигляді байтового рядка, що подається на вхід перетворення RotateLeft (<<< в ДСТУ 7624:2014). Цей рядок циклічно зсувається вліво на $2 \cdot N_b + 3$ байт, і згодом знову представляється у вигляді стану, що використовується як цикловий ключ з непарним індексом, тобто $k_{2i+1} = k_{2i} <<< (2 \cdot N_b + 3)$.

Таким чином, байтовий рядок $b_0, b_1, ..., b_{8 \cdot N_b - 1}$ після перетворення набуває вигляду $b_{2 \cdot N_b + 3}, b_{2 \cdot N_b + 4}, ..., b_{8 \cdot N_b - 1}, b_0, b_1, ..., b_{2 \cdot N_b + 2}$.

```
void Kalyna_KeyExpansion(byte key[8 * Nk], byte Ksigma[8 * Nb],
byte roundkey[(Nr + 1)][8 * Nb]) {

byte initial_data[ 8, Nk ] = key
byte state[ 8, Nb ]
byte kt_round[ 8, Nb ]

byte tmp_modification_value[ 8, Nb ] = 0x01, 0x00, 0x01, 0x00, …, 0x01, 0x00
byte round_key_index = 0

      while ( true ) {
            state = initial_data[ 0 .. 8 * Nb – 1 ]
            kt_round = Ksigma

            Add64RoundKey( kt_round, tmp_modification_value )

            Add64RoundKey( state, kt_round )

            Kalyna_S_boxes( state )
            ShiftRows( state )
            MixColumns( state )

            XORRoundKey( state, kt_round )

            Kalyna_S_boxes( state )
            ShiftRows( state )
            MixColumns( state )

            Add64RoundKey( state, kt_round )

            roundkey[ round_key_index ] = state

            if ( Nr == round_key_index ) break;

            if ( Nk > Nb ) {
                  round_key_index = round_key_index + 2;

                  ShiftLeft( tmp_modification_value )
                  state = initial_data[ 8 * Nb .. 8 * Nk – 1 ]

                  kt_round = Ksigma
                  Add64RoundKey( kt_round, tmp_modification_value )

                  Add64RoundKey( state, kt_round )
```

```
                Kalyna_S_boxes( state )
                ShiftRows( state )
                MixColumns( state )

                XORRoundKey( state, kt_round )

                Kalyna_S_boxes( state )
                ShiftRows( state )
                MixColumns( state )

                Add64RoundKey( state, kt_round )

                roundkey[ round_key_index ] = state

                if ( Nr == round_key_index ) break;
            }

            round_key_index = round_key_index + 2;
            ShiftLeft ( tmp_modification_value )
            Rotate( initial_data )
}
}
```

Рис. 18. Псевдокод процедури формування циклових ключів з парними індексами

Формування циклових ключів з непарними індексами виконується за допомогою процедури Kalyna_KeyExpansion_odd (рис. 19).

```
void Kalyna_KeyExpansion_odd ( byte roundkey[ Nr + 1 ][ 8 * Nb ] )
{
      for( round_key_index = 1 to Nr-1 step 2 ) {
                roundkey[ round_key_index ] = roundkey[ round_key_index – 1 ]
                RotateLeft( roundkey[ round_key_index ], 2 * Nb + 3 )
      }
}
```

Рис. 19. Псевдокод процедури формування циклових ключів з непарними індексами

Залежність константи, що визначає значення зсуву вліво циклового ключа з парним індексом від розміру блока, наведена у табл. 5.

Таблиця 5

Константа, що визначає значення зсуву вліво циклового ключа з парним індексом в залежності від розміру блока

Розмір блоку, біт (байт)	Зсув вліво(байт)
128 (16)	7
256 (32)	11
512 (64)	19

Висновки

„Калина” є симетричним блоковим шифром, який визначений у національному стандарті України ДСТУ 7624:2014, із підтримкою розміру блоку і довжини ключа 128, 256 і 512 бітів (довжина ключа дорівнює розміру блоку або в два рази перевищує його). „Калина” побудована на основі Rijndael-подібної SPN-структури, але, на відміну від інших алгоритмів, в новому стандарті застосовуються:

– попереднє і прикінцеве забілювання (pre- and postwhitening) із використанням модульного додавання (2^{64});

– чотири S-блока (замість одного) із вдосконаленими криптографічними властивостями для забезпечення додаткового запасу стійкості шифра;

– збільшений розмір МДВ-матриці лінійного перетворення, що покращує криптографічні властивості і є оптимальним для швидкодіючої реалізації на сучасних 64-бітових платформах;

– нова односпрямована схема розгортання циклових ключів, що забезпечує як додаткову стійкість до низки методів криптографічного аналізу, так і достатню швидкодію перетворення.

Розгорнутий альтернативний опис блокового шифру „Калина” відповідає визначеному в ДСТУ 7624:2014, але використовує інші позначення, традиційні для галузі комп’ютерних наук.

Список літератури: 1. *ГОСТ 28147–89.* Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. – Введ. 01–07–1990. – М. : Изд-во стандартов, 1989. – 28 с. 2. *Advanced* Encryption Standard (AES) [Electronic resource] : FIPS PUB 197. – 2001 – Mode of access : www. URL: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf. 3. *Courtois, Nicolas T.* Security evaluation of GOST 28147-89 in view of international standardisation. Cryptologia 36.1 (2012): 2-13. 4. *Державна* служба спеціального зв'язку та захисту інформації України, Інститут кібернетики імені В.М. Глушкова Національної академії наук України. Положення про проведення відкритого конкурсу криптографічних алгоритмів. [Electronic resource] Mode of access : www. URL: http://www.dstszi.gov.ua/dstszi/control/uk/publish/printable_article?art_id=48383. 5. *ДСТУ 7624:2014.* Інформаційні технології. Криптографічний захист інформації. Алгоритм симетричного блокового перетворення. – Введ. 01–07–2015. – К. : Мінекономрозвитку України, 2015.

Додаток

Таблиці підстановки алгоритму «Калина»

Підстановка S0															
A8	43	5F	06	6B	75	6C	59	71	DF	87	95	17	F0	D8	09
6D	F3	1D	CB	C9	4D	2C	AF	79	E0	97	FD	6F	4B	45	39
3E	DD	A3	4F	B4	B6	9A	0E	1F	BF	15	E1	49	D2	93	C6
92	72	9E	61	D1	63	FA	EE	F4	19	D5	AD	58	A4	BB	A1
DC	F2	83	37	42	E4	7A	32	9C	CC	AB	4A	8F	6E	04	27
2E	E7	E2	5A	96	16	23	2B	C2	65	66	0F	BC	A9	47	41
34	48	FC	B7	6A	88	A5	53	86	F9	5B	DB	38	7B	C3	1E
22	33	24	28	36	C7	B2	3B	8E	77	BA	F5	14	9F	08	55
9B	4C	FE	60	5C	DA	18	46	CD	7D	21	B0	3F	1B	89	FF
EB	84	69	3A	9D	D7	D3	70	67	40	B5	DE	5D	30	91	B1
78	11	01	E5	00	68	98	A0	C5	02	A6	74	2D	0B	A2	76
B3	BE	CE	BD	AE	E9	8A	31	1C	EC	F1	99	94	AA	F6	26
2F	EF	E8	8C	35	03	D4	7F	FB	05	C1	5E	90	20	3D	82
F7	EA	0A	0D	7E	F8	50	1A	C4	07	57	B8	3C	62	E3	C8
AC	52	64	10	D0	D9	13	0C	12	29	51	B9	CF	D6	73	8D
81	54	C0	ED	4E	44	A7	2A	85	25	E6	CA	7C	8B	56	80

Підстановка S1															
CE	BB	EB	92	EA	CB	13	C1	E9	3A	D6	B2	D2	90	17	F8
42	15	56	B4	65	1C	88	43	C5	5C	36	BA	F5	57	67	8D
31	F6	64	58	9E	F4	22	AA	75	0F	02	B1	DF	6D	73	4D
7C	26	2E	F7	08	5D	44	3E	9F	14	C8	AE	54	10	D8	BC
1A	6B	69	F3	BD	33	AB	FA	D1	9B	68	4E	16	95	91	EE
4C	63	8E	5B	CC	3C	19	A1	81	49	7B	D9	6F	37	60	CA
E7	2B	48	FD	96	45	FC	41	12	0D	79	E5	89	8C	E3	20
30	DC	B7	6C	4A	B5	3F	97	D4	62	2D	06	A4	A5	83	5F
2A	DA	C9	00	7E	A2	55	BF	11	D5	9C	CF	0E	0A	3D	51
7D	93	1B	FE	C4	47	09	86	0B	8F	9D	6A	07	B9	B0	98
18	32	71	4B	EF	3B	70	A0	E4	40	FF	C3	A9	E6	78	F9
8B	46	80	1E	38	E1	B8	A8	E0	0C	23	76	1D	25	24	05
F1	6E	94	28	9A	84	E8	A3	4F	77	D3	85	E2	52	F2	82
50	7A	2F	74	53	B3	61	AF	39	35	DE	CD	1F	99	AC	AD
72	2C	DD	D0	87	BE	5E	A6	EC	04	C6	03	34	FB	DB	59
B6	C2	01	F0	5A	ED	A7	66	21	7F	8A	27	C7	C0	29	D7

Підстановка S2															
93	D9	9A	B5	98	22	45	FC	BA	6A	DF	02	9F	DC	51	59
4A	17	2B	C2	94	F4	BB	A3	62	E4	71	D4	CD	70	16	E1
49	3C	C0	D8	5C	9B	AD	85	53	A1	7A	C8	2D	E0	D1	72
A6	2C	C4	E3	76	78	B7	B4	09	3B	0E	41	4C	DE	B2	90
25	A5	D7	03	11	00	C3	2E	92	EF	4E	12	9D	7D	CB	35
10	D5	4F	9E	4D	A9	55	C6	D0	7B	18	97	D3	36	E6	48
56	81	8F	77	CC	9C	B9	E2	AC	B8	2F	15	A4	7C	DA	38
1E	0B	05	D6	14	6E	6C	7E	66	FD	B1	E5	60	AF	5E	33
87	C9	F0	5D	6D	3F	88	8D	C7	F7	1D	E9	EC	ED	80	29
27	CF	99	A8	50	0F	37	24	28	30	95	D2	3E	5B	40	83
B3	69	57	1F	07	1C	8A	BC	20	EB	CE	8E	AB	EE	31	A2
73	F9	CA	3A	1A	FB	0D	C1	FE	FA	F2	6F	BD	96	DD	43
52	B6	08	F3	AE	BE	19	89	32	26	B0	EA	4B	64	84	82
6B	F5	79	BF	01	5F	75	63	1B	23	3D	68	2A	65	E8	91
F6	FF	13	58	F1	47	0A	7F	C5	A7	E7	61	5A	06	46	44
42	04	A0	DB	39	86	54	AA	8C	34	21	8B	F8	0C	74	67

Підстановка S3															
68	8D	CA	4D	73	4B	4E	2A	D4	52	26	B3	54	1E	19	1F
22	03	46	3D	2D	4A	53	83	13	8A	B7	D5	25	79	F5	BD
58	2F	0D	02	ED	51	9E	11	F2	3E	55	5E	D1	16	3C	66
70	5D	F3	45	40	CC	E8	94	56	08	CE	1A	3A	D2	E1	DF
B5	38	6E	0E	E5	F4	F9	86	E9	4F	D6	85	23	CF	32	99
31	14	AE	EE	C8	48	D3	30	A1	92	41	B1	18	C4	2C	71
72	44	15	FD	37	BE	5F	AA	9B	88	D8	AB	89	9C	FA	60
EA	BC	62	0C	24	A6	A8	EC	67	20	DB	7C	28	DD	AC	5B
34	7E	10	F1	7B	8F	63	A0	05	9A	43	77	21	BF	27	09
C3	9F	B6	D7	29	C2	EB	C0	A4	8B	8C	1D	FB	FF	C1	B2
97	2E	F8	65	F6	75	07	04	49	33	E4	D9	B9	D0	42	C7
6C	90	00	8E	6F	50	01	C5	DA	47	3F	CD	69	A2	E2	7A
A7	C6	93	0F	0A	06	E6	2B	96	A3	1C	AF	6A	12	84	39
E7	B0	82	F7	FE	9D	87	5C	81	35	DE	B4	A5	FC	80	EF
CB	BB	6B	76	BA	5A	7D	78	0B	95	E3	AD	74	98	3B	36
64	6D	DC	F0	59	A9	4C	17	7F	91	B8	C9	57	1B	E0	61

Харківський національний
університет радіоелектроніки

Надійшла до редколегії 30.04.2015