Дәріс 5. Гаммалау режимі

Жоспар:
1. Гаммалау режимінде шифрлау
2. AES шифры
3. Кілттерді генерациялау алгоритмі (Key Schedule)
4. Кеңейту (жоспарлау) кілті
5. Раундтық кілтті таңдау

Гаммалау режимінде шифрлау

Гаммалау режимінде шифрлау алгоритмін жүзеге асыратын криптографиялық схема 2.8 – суретте көрсетілген.

64 биттік блоктарға бөлінген ашық деректер гаммалау режимінде 64 биттік блоктар шығаратын $Г_ш$ шифрының гаммасы бар $СМ_5$ қосындысында 2 Модуль бойынша разрядты қосу арқылы шифрланады, яғни:

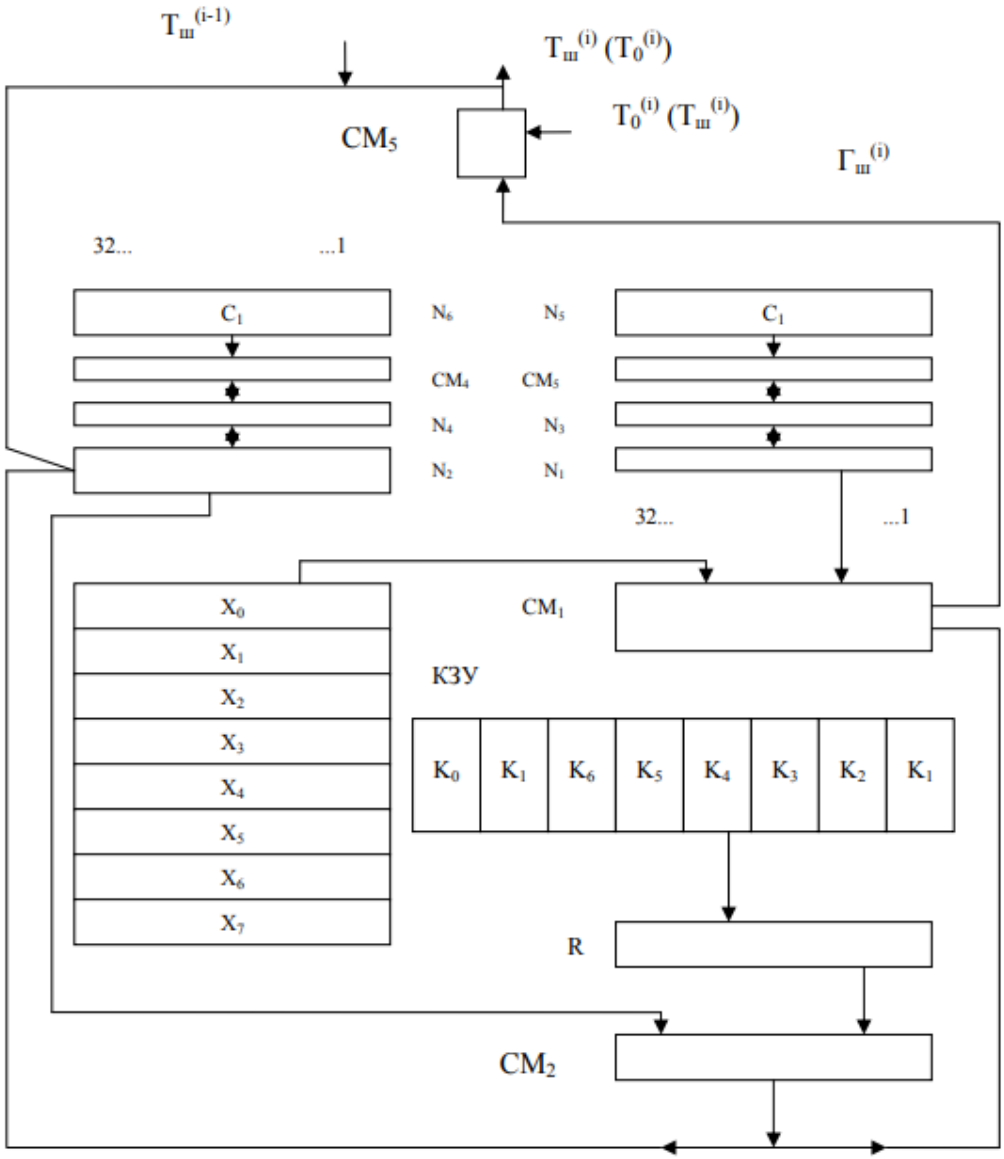
$$Г_Ш=(Г_Ш^{(1)}, Г_Ш^{(2)}, \ldots, Г_Ш^{(M-1)}, Г_Ш^{(M)}),$$

мұндағы M–шифрланған деректер көлемімен анықталады.

$Г_ш$ i-ші 64 разрядты блок, $i=1\div M$. $T^{(M)}$ блогындағы екілік цифрлардың саны 64-тен аз болуы мүмкін, бұл ретте шифрлық гамманың $Г_ш^{(M)}$ блогынан шифрлау үшін пайдаланылмайтын бөлігі жойылады.

КСҚ-ға кілттің 256 биті енгізілген. 64 биттік екілік тізбектер (синхрондау хабарламалары). Шифрлық гамманың M блоктарын кейіннен шығару үшін осы жетектерді бастапқы толтыру болып табылады.

$$S=(S_1, S_2,..., S_{64}),$$



2.8 – сурет. Қарапайым ауыстыру режиміндегі дешифрлау алгоритмінің криптографиялық схемасы

Синхрондау хабары N_1 және N_2 келесідей енгізіледі:

1) S_1 мәні N_1 бірінші цифрына, S_2 мәні N_1 екінші цифрына және т.б.

2) N_1 санының 32–ші цифрына S_{32} мәні, S_{32} бірінші цифрына S_{33} мәні, N_2-нің екінші цифрына S_{34} мәні және т.б.

3) S_{64} мәні N_2-нің 32 разрядына енгізіледі.

N_1 және N_2 дискілерінің бастапқы мазмұны (S синхрондау хабары) қарапайым ауыстыру режимінде шифрланған. Шифрлау нәтижесі $A(S) = (Y_0, Z_0)$ 32биттік N_3 және N_4 дискілеріне қайта жазылады, осылайша N_1 толтырғышы N_3-ке, ал N_2 толтырғышы N_4-ке қайта жазылады.

N_4 дискінің толтырылуы CM_4 сумматорында N_6 дискіден 32 разрядты C_1 тұрақтысы бар модуль бойынша $(2^{32}–1)$ жинақталады, нәтиже N_4 жазылады. Екі бүтін a, b ($0 * a, b * 2^{32}–1$) келесі ережеге сәйкес $2^{32} – 1$ модуліне қосылды:

$$a*b = a + b, \text{ егер } a + b < 2^{32},$$

$$a*b = a + b - 2^{32} + 1, \text{ егер } a + b < 2^{32},$$

N_3 қоймасын толтыру СМ3 қосқышында 2^{32}модуль бойынша жинақталған N_5 дискідегі 32 биттік C_2 тұрақтысында нәтиже N_3 жазылады.

C_1 және C_2 тұрақтыларының келесі мағыналары бар:

C_1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0;
C_2 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1.

N_6 және N_5 жетектерінің разрядтарының реті, оларда C_1 және C_2 тұрақтылары келесідей:

N_6 (C_1) 32 31 30 29 28 27 … 1;
N_5 (C_2) 32 31 30 29 28 27 … 1.

N_3 толтырғышы N_1-ге қайта жазылады, ал N_4 толтырғышы қайта жазылады және N_3 және N_4 толтыру сақталады.

N_1 және N_3 толтырғыштары алгоритм талаптарына сәйкес қарапайым ауыстыру режимінде шифрланған. Шифрлау нәтижесінде алынған N_1 және N_2 толтыру $Г_{ш}^{(1)}$ гамма шифрінің бірінші 64 разрядты блогын құрайды, ол $СМ_5$ сумматорында 2 разрядтық модуль бойынша жинақталған.

AES шифры. Деректер блогының пішімі және айналымдар саны

RIJNDAEL 128 биттік деректер блоктарымен және 128, 192 немесе 256 биттік кілт ұзындығымен жұмыс істейтін симметриялы блок шифры. Кілт ұзындығына байланысты стандарт сәйкесінше AES-128, AES-192 немесе AES-256 деп аталады.

Келесі белгілерді енгізейік:

Nb – кіріс блогындағы 32 разрядты сөздердің саны, *Nb* =4;

Nk – шифрлау кілтіндегі 32 разрядты сөздердің саны, *Nk* =4,

Nr – шифрлау айналымдарының саны, *Nb* және *Nk* функциясы ретінде, *Nr*=10, 12, 14.

Енгізу (*input*), аралық (*state*) және шығыс (*output*) нәтижелер криптографиялық алгоритм ішінде орындалатын түрлендірулер күйлер деп аталады. Күй 4•*Nb* матрицасы ретінде ұсынылуы мүмкін, оның элементтері байт (әрқайсысы *Nb* байттың төрт жолы) келесі ретпен $S00, S10, S20, S30, S01, S11, S21, S31$, ... орындалады.

2.9 – суретте «Шаршы» архитектурасы деп аталатын мұндай көрініс көрсетеді.

Input

in0	in4	in8	in12
in0	in0	in0	in0
in0	in0	in0	in0

State

s0,0	s0,1	s0,2	s0,2
s1,0	s1,1	s1,2	s1,3
s2,0	s2,1	s2,2	s2,3
s3,0	s3,1	s3,2	s3,3

Output

out0	out4	out8	out12
out1	out5	out9	out13
out2	out6	out10	out14
out3	out7	out11	out15

2.9 – сурет. «Шаршы» архитектурасы

4**Nb* матрицасы ретінде блокты ұсынудың мысалы, шифрлау және шифрды шешу процестерінің басында кіріс деректерінің массиві

$$in0, in1, \dots, in15$$

ережеге сәйкес *State* массивіне түрлендіріледі

$$S[r, c] = in[r + 4c],$$

мұндағы $0 \leq r < 4$ и $0 \leq c < Nb$. Алгоритмнің соңында кері түрлендіру орындалады.

$$out[r + 4c] = s[r, c],$$

мұндағы $0 \leq r < 4$ және $0 \leq c < Nb$. Шығару күй байттарынан бірдей ретпен алынады.

Әрбір күй бағанындағы төрт байт 32 биттік сөзді білдіреді. Егер r күй жолының нөмірі болса, онда ол да осы сөздегі әрбір байттың индексі болып табылады. Осылайша, күйді 32 биттік сөздердің бір өлшемді массиві ретінде ұсынуға болады.

$$w0, \dots, wNb,$$

мұндағы c күй бағанының нөмірі осы массивтегі индекс болып табылады. Сонда блокты келесідей көрсетуге болады

*w*0=*s*0,0, *s*1,0, *s*2,0, *s*3,0 *w*2=*s*0,2, *s*1,2, *s*2,2, *s*3,2

$w1=s0,1, s1,1, s2,1, s3,2 \quad w3=s0,3, s1,3, s2,3, s3,3$

шифрлау кілті, *State* массиві сияқты, төрт жолдан тұратын тікбұрышты массив ретінде ұсынылған. Бұл массивтің бағандарының саны Nk.

AES алгоритмі үшін раундтардың саны Nr басында Nk мәніне байланысты анықталады. 2.9 – кестеде Nr мәнінің Nk және Nb тәуелділігі көрсетілген.

2.9 – кесте. Nr мәнінің Nk және Nb–ге тәуелділігі

Шифры	Nk	Nb	Nb
AES-128	4	4	10
AES -192	6	4	12
AES -256	8	4	14

Келесі белгілерді енгізейік:

SubBytes() – байттарды ауыстыру. Бекітілген 8х256 ауыстыру кестесін (*affain map table*) пайдалана отырып, күй блоктарында (*S–Box*) байт–байт сызықты емес ауыстыру;

ShiftRows() – жолдарды ауыстыру. *State* жиымының жолдарының әртүрлі байт санына циклдік ығысуы;

MixColumns() – бағандарды араластыру. $GF(2^8)$ бойынша полином ретінде қарастырылатын күй бағандарын көбейту;

AddRoundKey() – дөңгелек кілтпен қосу. Биттік *XOR* – ағымдағы кеңейтілген кілт фрагментіндегі мазмұн әрекетін күй.

Псевдокодта шифрлау операциясы келесідей болады:

*Cipher (byte in [4*Nb], byte out [4*Nb], word w [Nb*(Nr+1)]) begin*
byte state [4, Nb] state=in
AddRoundKey (state, w[0, Nb–1]) for round=1 step 1 to Nr–1
Subbytes (state) ShiftRows(state) MixColumns (state)
*AddRoundKey (state, w[round*Nb, (round+1)*Nb–1]) end for*
SubBytes (state) ShiftRows(state)
*AddRoundKey (state, w[Nr*Nb, (Nr+1)*Nb–1]) out=state*
end

State массивін кіріс деректер элементтерімен толтырғаннан кейін оған *AddRoundKey()* түрлендіруі қолданылады. Әрі қарай, *Nk* мәніне байланысты күй массиві 10, 12, 14 рет айналмалы түрлендіруден өтеді, ал соңғы раунд біршама қысқарады, оның құрамында *MixColumns()* түрлендіруі жоқ. Сипатталған операциялар тізбегінің шығысы шифрленген мәтін болып табылатын, AES шифрлау функциясының нәтижесі.

Кері шифрлау функциялары. *AES* алгоритмінің спецификациясы шифрлау функциясын жүзеге асырудың екі түрін ұсынады, олар бір бірінен шифрлау функциясының кері түрлендірулерінің түрлендіру реттілігімен және кілтті жоспарлау реттілігімен ерекшеленеді.

Келесі белгілерді енгізейік:

InvSubBytes() – кері *SubBytes* байтты ауыстыру, бекітілген 8x256 (*inverse affain map*) ауыстыру кестесін пайдаланып күй блоктарында байт және байт сызықты емес ауыстыру;

InvShiftrows() – кері жолды жылжыту, *ShiftRows()* күй массивінің жолдарын әртүрлі байт санына циклдік ауыстыру;

InvMixColumns() – баған мәндерін қалпына келтіру. *GF* (2^8) бойынша полином ретінде қарастырылатын күй бағандарын көбейту.

Шифрды кері шешу функциясы.

Егер *Subbytes(), ShiftRows(), MixColumns()* және *AddRoundKey()* операцияларының орнына кері ретпен кері түрлендірулер жасалса, кері шифрлау (шифрды ашу) функциясын құруға болады. Бұл жағдайда раундтық кілттерді пайдалану тәртібі шифрлау кезінде қолданылатынға кері болып табылады. Псевдокодта декодтау процесі келесідей:

```
   InvCipher (byte in[4*Nb], byte out [4*Nb],
word w[Nb*(Nr+1)])
   begin
      byte state
      [4, Nb]
      state=in
      AddRoundKey (state, dw [Nr*Nb,
      (Nr+1)*Nb–1]) for round=Nr–1 step –1
      downto 1
         InvShiftRows
         (state)
         InvSubBytes
         (state)
        AddRoundKey (state, w [round*Nb,
       (round+1)*Nb–1]) InvMixColumns (state)
      end for
      InvShiftRows
      (state)
      InvSubBytes
      (state)
      AddRoundKey (state, w [0, Nb–1])
      out=state
   end
```

Тікелей транскрипция функциясы шифрланған мәтіннен бастапқы деректерді қалпына келтіруге мүмкіндік беретін кері түрлендірулер тізбегін орындайды. Псевдокодта декодтау процесі келесідей:

```
EqInvCipher (byte in [4*Nb], byte out [4*Nb], word dw [Nb*
(Nr+1)])
```

```
 begin
    byte state [4, Nb]
state=in
AddRoundKey (state, dw [Nr*Nb, (Nr+1)*Nb–1])
for round=Nr–1 step –1 downto 1
     InvSubBytes (state)
     InvShiftRows (state)
     InvMixColumns (state)
     AddRoundKey (state, dw [round*Nb, (round+1)*Nb–1])
end for
InvSubBytes (state)
InvShiftRows (state)
AddRoundKey (state, dw [0, Nb–1])
out state
end
```

Жоғарыда сипатталған кері шифрлау алгоритмі операцияларды тікелей шифрлау алгоритміне кері ретпен орындайды, бірақ бірдей кеңейтілген кілт параметрлерін қолданады. Алайда, кілтті жоспарлау ретін өзгерту арқылы альтернативті алгоритм, тікелей декодтау алгоритмін құруға болады.

Екі негізгі қасиет осы тәсілді жүзеге асыруға мүмкіндік береді:

1) *Subbytes ()* және *ShiftRows ()* функцияларын қолдану тәртібінің тәуелсіздігі. *Subbytes()* және *Invsubbytes()* функциялары бөлек байттармен жұмыс істейді, ал ShiftRows() және *InvShiftRows()* мәндерді өзгертпестен олардың орналасуын өзгертеді. Сондықтан олардың орындалу ретін нәтижеге әсер етпей өзгертуге болады.

2) кіріс деректеріне қатысты *Mixcolumns ()* операциясының Сызықтығы.

Бұл қасиет келесі теңдікте көрінеді:

$$MixColumns() (State \oplus roundKey) == MixColumns() (State) \oplus MixColumns() (RoundKey)$$

мұндағы *Invsubbytes()* және *InvShiftRows()* операцияларын алгоритмнің дұрыстығын жоғалтпай қайта реттеуге мүмкіндік береді. Сонымен қатар, егер кеңейтілген кілт бағандары *InvMixColumns()* функциясымен алдын ала өңделсе, *AddRoundKey()* және *InvMixColumns()* функцияларын кері тәртіпте қолдануға болады.

Осылайша, шифрлау кезіндегідей операциялар тізбегін қолдана отырып, декодтауды тиімдірек жүзеге асыру мүмкін болады. Бұл есептеу шығындарын азайту арқылы алгоритмді аппараттық және бағдарламалық қамтамасыз етуді жеңілдетеді.

Кілттерді генерациялау алгоритмі (Key Schedule)

Keys Schedule алгоритмі бастапқы шифрлау кілтінен дөңгелек кілттер жиынтығын жасау үшін қолданылады. Бұл процесс бастапқы кілтті күрделі

және тұрақты емес түрлендіру арқылы AES алгоритмінің криптографиялық төзімділігін қамтамасыз етеді.

Келесі белгілерді енгізейік:

Rcon[] – 32 разрядты дөңгелек тұрақтылар массиві;

RotWord() – келесі ережеге сәйкес кіріс 4 байт сөзді шығыс сөзге циклдік қайта реттеу операциясы $[a_0, a_1, a_2, a_3] \rightarrow [a_1, a_2, a_3, a_0]$;

SunWord() – S Box көмегімен 4 байт сөздегі операцияны әрбір байт бойынша ауыстыру, эксклюзивті немесе *XOR* операциясы.

Раундтық кілттер шифрлау кілтінен кілтті шығару алгоритмі арқылы алынады. Ол екі компоненттен тұрады: кілттерді кеңейту (*Key Expansion*) және раунд кілттерін таңдау (*Round Key selection*). Алгоритмнің негізгі принциптері келесідей:

1) раундтық кілт биттерінің жалпы саны блок ұзындығының айналымдар санына 1 плюс көбейтіндісіне тең;

2) шифрлау кілті *кеңейту* кілтіне (*Expansion Key*) кеңейтілген;

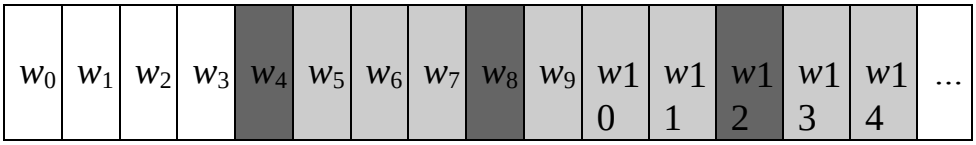
3) раундтық кілттер кеңейтілген кілттерден келесідей алынады: бірінші раунд кілтінде бірінші *Nb* сөздері, екіншісінде келесі *Nb* сөздері және т.б.

```
KeyExpansion (byte key [4*nk], word w [Nb,*(Nr+1)], Nk)
begin
    word temp
    i=0
    while (i<Nk)
        w[i]=word (key [4+i], key [4*i+1}, key [4*i+2], key
    [4*i+3]) end while
    i=Nk
    while      (i<Nb*(Nr–1)]
        temp=w[i+1]
        if (i mod Nk=0)
            temp=SubWord (RotWord (temp)) xor Rcon
        [i/Nk]
            else if (Nk>6 and I mod Nk=4)
              temp=Subword(temp)
        end if
        w[i]=w[i–Nk] xor temp i=i+1
    end while
end
```

Кеңейту (жоспарлау) кілті

AES алгоритміндегі кілтті жоспарлау (Key Expansion) бір бастапқы кілттен дөңгелек кілттер жиынтығын құру мақсатында жүзеге асырылады. Бұл кілттер шифрлау мен шифрды шешудің әр кезеңінде қолданылады

2.10 – суреттегі кеңейтілген кілт сызықтық массив болып табылады $w[i]$, $4(Nr + 1)$, байт сөзден тұрады, $i = 0{,}4(Nr + 1)$.



2.10 – сурет. *Nk*=4 үшін раундтық кілтті кеңейту және таңдау процедуралары

2.10 – суретте SubWord() және RotWord() функцияларын пайдаланбай жасалған кеңейтілген кілттің сөздері ашық сұр түспен ерекшеленген. Кеңейтілген кілттің сөздері қою сұр түспен бөлектеледі, оны есептеу үшін SubWord() және RotWord() түрлендірулері қолданылады).

Бірінші *Nk* сөздерінде шифрлау кілті бар.Әр келесі $w[i]$ сөзі *XOR* алдыңғы $w[i-1]$ сөзін және *Nk* сөзін бұрынғы позицияларды түрлендіру арқылы алынады, яғни

$$[i-Nk]: w[i] = w[i-1] \circ w[i-Nk].$$

Орны *Nk* еселігі болатын сөздер үшін *XOR* түрлендіру алдында түрлендіру $w[i-1]$ қолданылады, содан кейін *Rcon[i]* дөңгелек тұрақтысы қосылады. Трансформация екі қосымша *SubWord()* және *RotWord()* функциялары арқылы жүзеге асырылады. *Rcon[j]* мәні 2^{j-1}. Бұл жағдайда $w[i]$ мәні өрнекпен анықталады

$$w[i] = SubWord\ (RotWord\ (w[i-1])) \circ w[i-Nk].$$

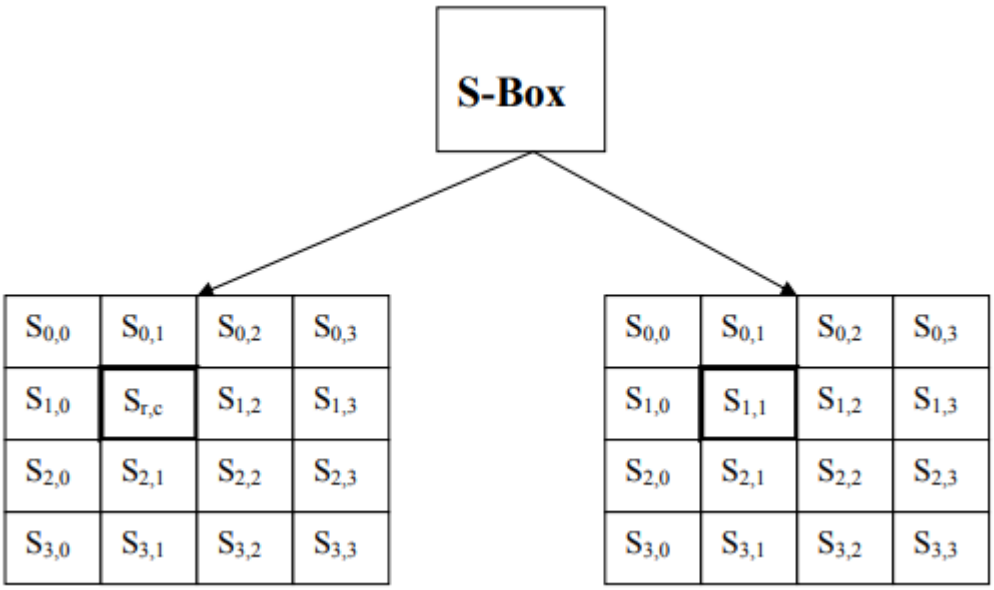
Раундтық кілтті таңдау

i–ші раунд кілті $W[Nb \bullet i]$-ден $W[Nb \bullet (i+1)]$ аралығындағы кеңейтілген кілттер массивінің сөздерінен таңдалады. Шифрлау функциясы үшін кеңейтілген кілт суретте көрсетілген алгоритм арқылы жасалады, 2.11–суретте көрсетілген. Кері шифрды шешудің кеңейтілген функциясы үшін бірдей кілт пайдаланылады, бірақ соңғы айналымдағы шифрлау кілтінен бастап кері тәртіпте.

Алға шифрды шешу функциясы сәл өзгертілген кілтті жоспарлау алгоритмін пайдаланады. Кеңейтілген кілтті генерациялау кезінде жоспарлау процедурасына соңында қосымша түрлендіруді қосу керек, ал кеңейтілген кілт шифрлау кезіндегідей реттілікпен тікелей шифрды шешу кезінде қолданылады.

```
for i=0 step 1 to
    (Nr+1)*Nb–1 dw
    [i]=w[i]
end for
for round =1 step 1 to Nr–1
```

*InvMixColumns (dw[round*Nb, (round+1)*Nb–1]) end for*



2.11 – сурет. S-Box ауыстыру кестесі арқылы күйді түрлендіру

Раундтық түрлендіру математикалық сипаттамасы 2 тарауда берілген *State* массивіне түрлендірулер қатарын дәйекті түрде қолданудан тұрады. Енді оның орындалуының егжей–тегжейлерін талқылаймыз.

Байтты ауыстыру 2 тарауда атап өтілгендей, *SubBytes()* арқылы күй массивінің байттарын сызықтық емес ауыстыру келесідей болады:

$$\lambda(f) \equiv (\,(X^4 + X^3 + X^2 + X + 1) \cdot f + X^6 + X^5 + X + 1)\; mod\; (X^8 + 1)$$

бұл өрнекті шифрлау процесі кезінде қайталанатын есептеулер орындау жүйесіне негізсіз есептеу жүктемесін түсіреді, сондықтан практикалық іске асыру үшін ең қолайлы шешім алдын ала есептелген *S - Box* кестесін пайдалану болып табылады. *{xy}* байтты түрлендіру кезіндегі *S - Box* логикасы 2.10 – кестеде он алтылық түрінде көрсетілген.

2.10 – кесте. S–Box байтты ауыстыру кестесі

	0	1	2	3	4	5	6	7	8	9	A	b	c	d	e	f
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	o5	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

2.10 – кестені пайдалану *SubBytes()* әрекетін $\lambda(f) = Sbox[f]$ массивінен қарапайым байт таңдауына дейін азайтады.

Шифрды шешу функциясы *SubBytes()* және *Inv SubBytes()* кері әрекетін пайдаланады, оның математикалық түсіндірмесі келесі 2.19 формуламен өрнектеледі.

$$\lambda(f) \equiv ((X^4 + X^3 + X^2 + X + 1) \cdot f + X^6 + X^5 + X + 1) \; mod \; (X^8 + 1) \qquad (2.19)$$

бірақ ол *S–Box* – $\lambda^{-1}(f) = InvSbox[f]$. Кері қойындысы арқылы алдыңғысы сияқты қарапайым орындалады. {ху} байтты түрлендіру кезіндегі оның жұмысының логикасы кестеде он алтылық формада 2.11-кестеде көрсетіледі.

2.11 – кесте. S–Box кері байтты ауыстыру кестесі

	0	1	2	3	4	5	6	7	8	9	a	B	c	d	e	F
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	76	d6	26	e1	69	14	63	55	21	0c	7d

2.11–кестеде шифрлау және дешифрлау функцияларындағы күйге байтты ауыстыру түрлендіруін қолдануды суреттейді.

Жолды ауыстыру түрлендіруінде *(ShiftRows)* күйдің соңғы үш жолы әртүрлі байт саны бойынша циклдік солға жылжиды. 1 жол C_1 байтқа, 2 жол C_2 байтқа және 3 жол C_3 байтқа жылжиды. Rijndael–дегі C_1, C_2 және C_3 жылжуларының мәні *Nb* блок ұзындығына байланысты. Олардың мәндері 2.12–кестеде берілген.

2.12 – кесте. Әр түрлі блок ұзындықтары үшін сызықты ауыстыру мөлшері

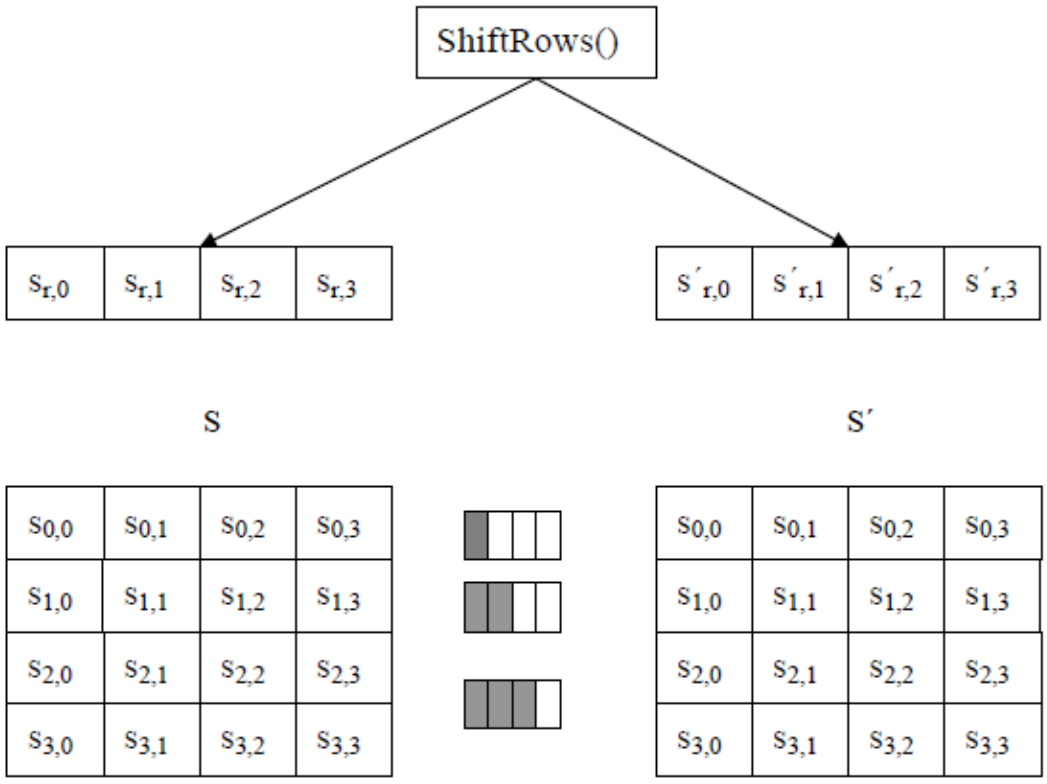
N_b	*shift* (0, *Nb*)=c	*shift* (0, *Nb*)=c	*shift* (0, *Nb*)=c	*shift* (0, *Nb*)=c	
4	0	1	2	3	**AES**
6	0	1	2	3	
8	0	1	3	4	

AES стандартында блок өлшемі 128 ғана анықталған бит, бізде $C_1 = 1, C_2 = 2$ және $C_3 = 3$.

Күйдің мазмұнын ауыстыру операциясын келесі (2.20) өрнекпен көрсетуге болады

$$S_{r,c} = S_{r,}\big(c + shift(r, Nb)\big) mod\ Nb \qquad (2.20)$$

$0 < r < 4$ және $0 \leq c < Nb$ үшін, мұндағы жылжу мәні $shift\,(r, Nb)$ жол нөміріне байланысты. 2.12–суретте *Shiftrows()* түрлендіруінің күйге әсерін көрсетеді.

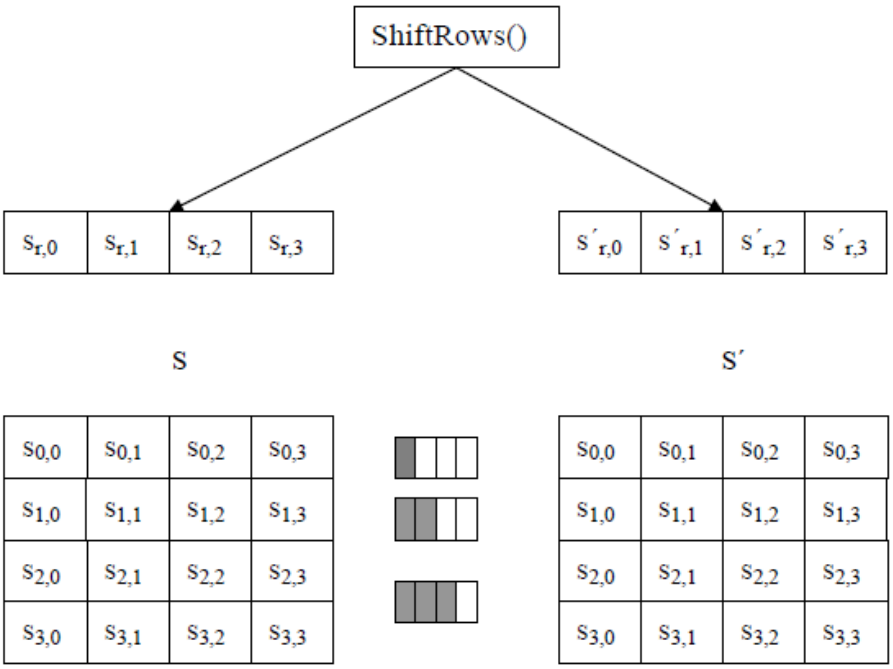


2.12 – сурет. Жолды ығыстыру арқылы дешифрлау функциясына түрлендіру

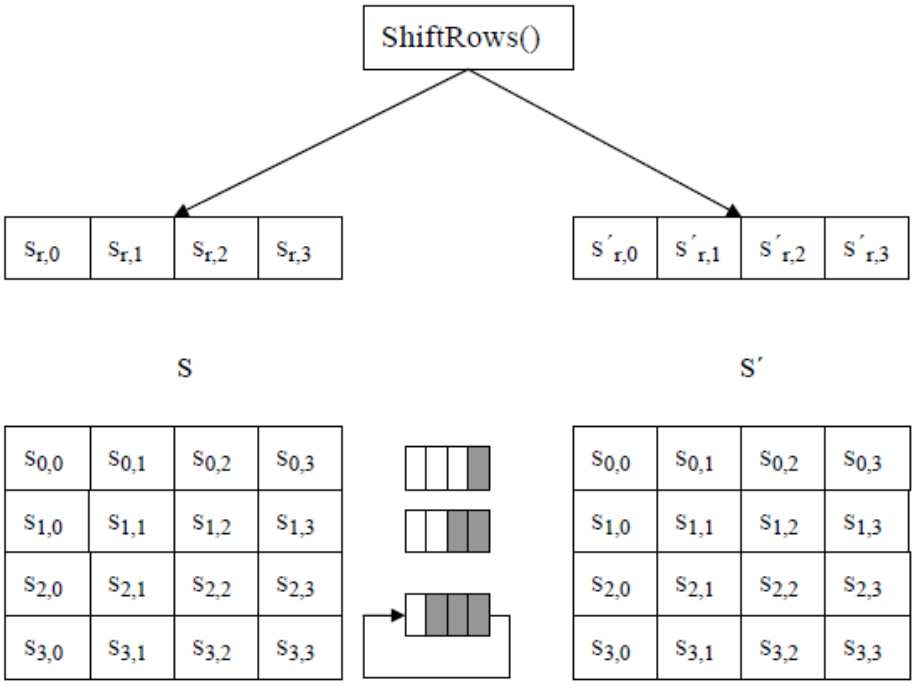
InvshiftRows кері жолды ауыстыру түрлендіруінде соңғы үш күй жолы әртүрлі байт саны бойынша циклді түрде оңға жылжиды. 1–жол C_1 байтқа, 2 жол C_2 байтқа, 3 жол C_3 байтқа ауыстырылады. C_1, C_2 және C_3 жылжуларының мәндері *Nb* блок ұзындығына байланысты. Олардың мәндері кестеде берілген мәндерге сәйкес келеді. 2.12 –кестеде *InvshiftRows()* күй мазмұнын кері жылжыту әрекеті (2.21) өрнекпен ұсынылған

$$S_{r,}\big(c + shift(r, nb)\big) mod\ Nb\ =\ S_{r,c} \qquad (2.21)$$

$0 < r < 4$ және $0 \leq c < Nb$ үшін, мұндағы *shift(r, Nb)* мәні *r* жол нөміріне байланысты. 2.13, 2.14 суреттер күй жолағын кері жылжыту түрлендірулерін суреттейді.



2.13 – сурет. Кері жолды шифрды шешу функцияларына ауыстыруды түрлендіру



2.14 – сурет. Кері жолды шифрды шешу функцияларына ауыстыруды түрлендіру

Бағандарды араластыруда (*Mixcolumns*) түрлендіруінде күй бағандары $GF(2^8)$ үстіндегі көпмүшеліктер ретінде қарастырылады. Олар трансформациядан өтеді

$$\mu(g) = \mathrm{c} \cdot g \; mod \; (Y^4 + 1),$$

мұндағы $c = (X, 1, 1, X + 1)$,яғни. олар $x^4 + 1$ модуліне *c(x)* көпмүшелігіне көбейтіледі, келесідей көрсетіледі

$$c(x) = \{03\}x^3 + \{01\}x + \{02\}.$$

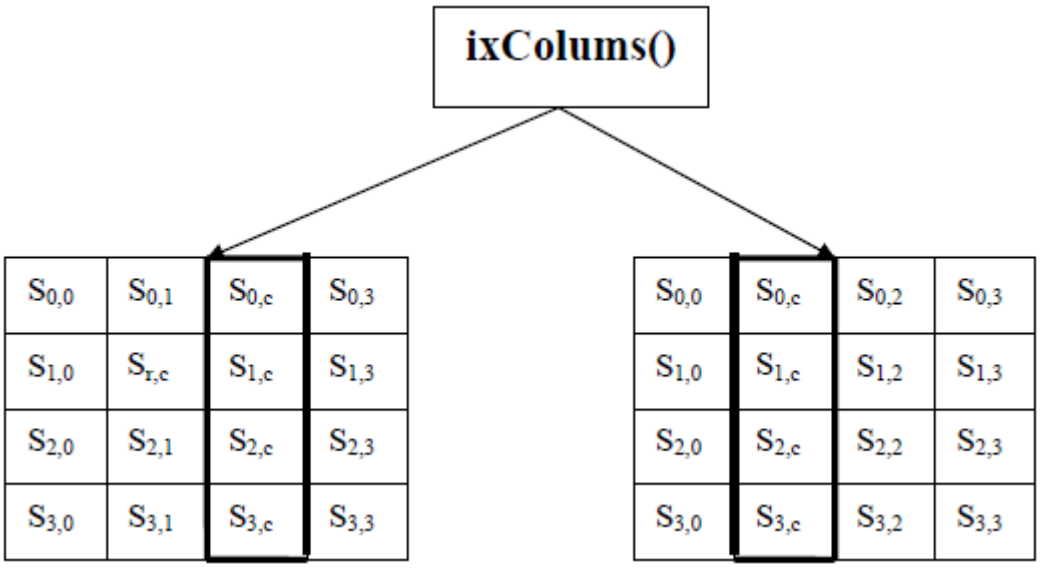
Бұл түрлендіруді матрицалық түрде келесідей көрсетуге болады:

$$\begin{Bmatrix} s'_{0c} \\ s'_{1c} \\ s'_{2c} \\ s'_{3c} \end{Bmatrix} = \begin{Bmatrix} 02\ 03\ 01\ 01 \\ 01\ 02\ 03\ 04 \\ 01\ 01\ 02\ 03 \\ 03\ 01\ 01\ 02 \end{Bmatrix},$$

$0 \le c \le c$, мұндағы *c State* массивінің баған нөмірі. Осы көбейту нәтижесінде s_{0c}, s_{1c}, s_{2c} және s_{3c} баған байттары сәйкесінше байттармен ауыстырылады.

$$s'_{0,c} = (\{02\} \cdot s_{0,c}) \circ (\{03\} \cdot s_{1,c}) \circ s_{2,c} \circ s_{3,c},$$
$$s'_{1\,c} = s_{0,c} \circ (\{02\} \cdot s_{1,c}) \circ (\{03\} \cdot s_{2,c}) \circ s_{3,c},$$
$$s'2, c = s_{0,c} \circ s_{1,c} \circ (\{02\} \cdot s_{2,c}) \circ (\{03\}) \cdot s_{3,c}),$$
$$s'3, c = (\{03\} \cdot s_{0,c}) \circ s_{1,c} \circ s_{2,c} \circ (\{02\}\} \cdot s_{3,c}).$$

Бұл әрекетті барлық төрт күй бағандарына қолдану *MixColumns*(*State*) ретінде белгіленеді. 2.15 – суретте күй бағанына *MixXolumns()* түрлендірулері көрсетілген.



2.15 – сурет. Араластыру операциясы күй бағандарына әсер етуі

InvMixColumns кері түрлендіруінде күй бағандары $GF(2^8)$ бойынша көпмүшеліктер ретінде қарастырылады, бірақ табиғи түрде кері түрлендіруге жатады (2.22) формуласы бойынша

$$v(g) = \mu^{-1}(g) \; = \; mod\ (Y^4 + 1), \qquad (2.22)$$

мұндағы

$$d = (X^3 + \ X^2 + \ X, X^3 + \ 1, X^3 + X^2 + \ 1, X^3 + \ X + 1)$$

яғни модуль бойынша $(x^4 + 1)\ d(x),$ көпмүшелігіне көбейтіледі, ол келесідей көрінеді

$$d(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}.$$

Оны матрицалық түрде келесідей көрсетуге болады:

$$\begin{Bmatrix} s'_{0c} \\ s'_{0c} \\ s'_{0c} \\ s'_{0c} \end{Bmatrix} = \begin{Bmatrix} 0e\ 0b\ 0d\ 09 \\ 0e\ 0b\ 0d\ 09 \\ 0e\ 0b\ 0d\ 09 \\ 0e\ 0b\ 0d\ 09 \end{Bmatrix},$$

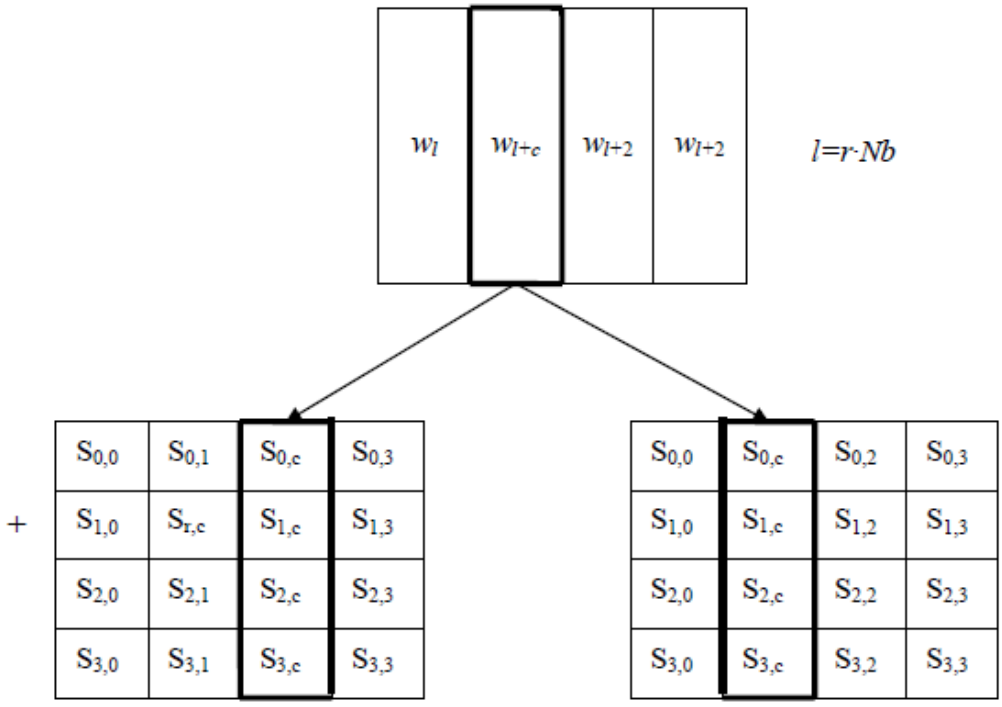
$0 \leq c \leq 3$, мұндағы c *State* массивінің баған нөмірі. Алынған нәтиже келесі байт болып табылады:

$$s'_{0c} = (\{0e\} \cdot s'_{0c}) \circ (\{0b\} \cdot s'_{1c}) \circ (\{0d\} \cdot s'_{2c}) \circ (\{09\} \cdot s'_{3c}),$$
$$s'_{1c} = (\{09\} \cdot s'_{0c}) \circ (\{0e\} \cdot s'_{1c}) \circ (\{0b\} \cdot s'_{2c}) \circ (\{0d\} \cdot s'_{3c}),$$
$$s'_{2c} = (\{0d\} \cdot s'_{0c}) \circ (\{09\} \cdot s'_{1c}) \circ (\{0e\} \cdot s'_{2c}) \circ (\{0b\} \cdot s'_{3c}),$$
$$s'_{3c} = (\{0b\} \cdot s'_{0c}) \circ (\{0d\} \cdot s'_{1c}) \circ (\{09\} \cdot s'_{2c}) \circ (\{0e\} \cdot s'_{3c}).$$

AddRoundKey () раундтық кілтін қосу. Бұл операцияда биттік *XOR* түрлендіруі арқылы күйге дөңгелек кілт қосылады. 32 биттік сөздердегі кілт ұзындығы *Nb* блок ұзындығына тең. Күйге раунд кілтін қосуды қамтитын түрлендіру келесі өрнек ретінде ұсынылады:

$$[s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c}] = [s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c}] \circ [w_{r(Nb+c)}],$$

мұндағы $0 \leq c \leq Nb$ және $0 \leq r \leq Nr$. 2.16–суретте бұл операцияның қалып–күй әсерін көрсетеді. Бұл түрлендіру *AddRoundKey*(*State*, *RoundKey*) ретінде белгіленеді.



2.16 – сурет. *AbbRoundKey*(*State*, *RoundKey*) күй бағандарында әрекет етеді

Тараудың соңында біз AES негізгі ерекшеліктерін тұжырымдаймыз:

1) Жаңа «Шаршы» архитектурасы ақпараттың тез таралуын және араласуын қамтамасыз етеді, бір раундта бүкіл кіріс блогы түрлендіріледі.

2) Байтты бағдарланған құрылым сегіз разрядты микроконтроллерлерде іске асыруға ыңғайлы.

3) Барлық раундтық түрлендірулер соңғы өрістердегі операцияларды білдіреді, бұл оларды әртүрлі платформаларда бағдарламалық және аппараттық түрде тиімді жүзеге асыруға мүмкіндік береді.

Ұсынылатын әдебиеттер тізімі

1. Гольдвассер С., Беллар М. Достижения в криптологии / Гольдвассер С. – М.: Триумф, 2016. – 513 с.

2. Шнайер, Брюс. Криптографическая методы защиты информации, 26-я ежегодная международная конференция по криптологии, Санта-Барбара, Калифорния, США, 20–24 августа 2022 г.

3. Фергюсон Н., Шнайер Б., Коно Т. Криптографическая инженерия: принципы проектирования и практическое применение. / Фергюсон Н. – М.: Пресс, 2022. – 416 с.