Atomic-AES

A Compact Implementation of the AES Encryption/Decryption Core

by

Subhadeep Banik

Joint work with Andrey Bogdanov, Francesco Regazzoni

Asian Symmetric Key Workshop, Nagoya, 2016

Sep 30, 2016

Introduction

Introduction

- Good Morning to all !!!
- Compact Implementation of AES ENC/DEC Circuit. Why ENC+DEC ?

  Many modes like CBC, ELmD, COPA need ENC+DEC access.
- Serial AES Circuit by Moradi et al. [Eurocrypt 11]

  One of the smallest at 2400 GE. Encrypt only.

  Description of structure, datapath and functioning.
- Atomic-AES : Both Encrypt and Decrypt supported.

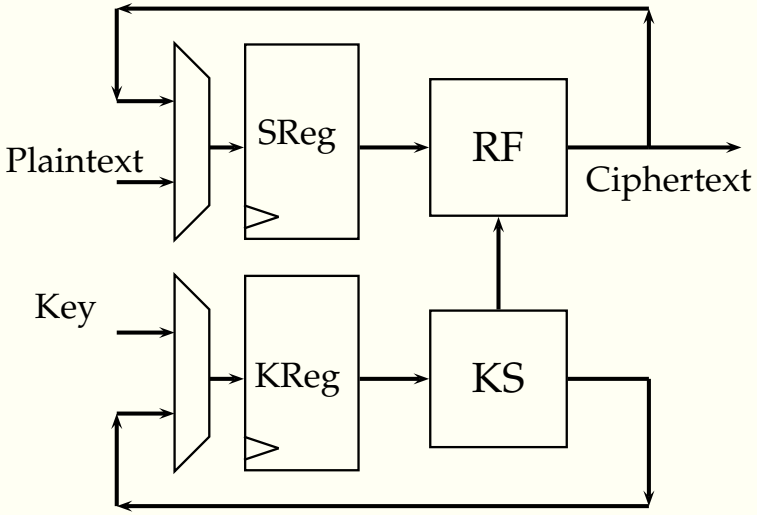
  Based on the Moradi circuit: 2645 GE: ENC/DEC latency: 226 cycles.

  Grain of Sand (Feldhofer et al. IEEE IS 05): 3400 GE, 1032/1165 cycles.

  Description of structure, datapath and functioning.

Serial Implementation

Serial vs Round based



- One round computed per clock cycle: No resource sharing.
- AES $\rightarrow$ 20 S-boxes per round !!
- Smallest: Canright [CHES 04], Boyar-Peralta [JOC 11]

  Forward S-box: 200GE approx: Hence 4000 GE for S-boxes alone!!

  AES Encryption ckt: 8000 GE.

Serial vs Round based

Tradeoffs

- Imagine AES circuit with only 8 S-box circuits.
- Atleast 3 cycles to do Substitution layer of one round.

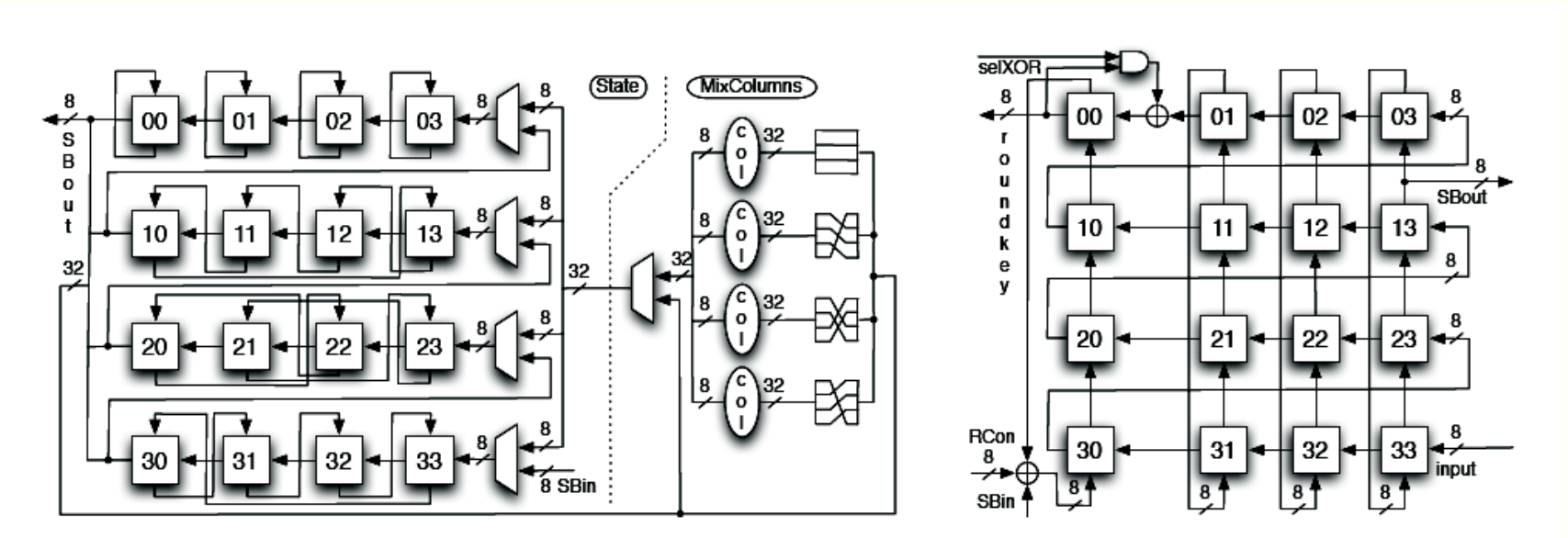
  Substitution layer for 8-bytes of state can be computed in one cycle.

  Atleast 3*10=30 cycles for one encryption $\rightarrow$ more latency.
- Most compact circuit: One S-box.

  Needs atleast 20*10=200 cycles for one encryption.

8-bit serial AES circuit (Moradi et al Eurocrypt 11)

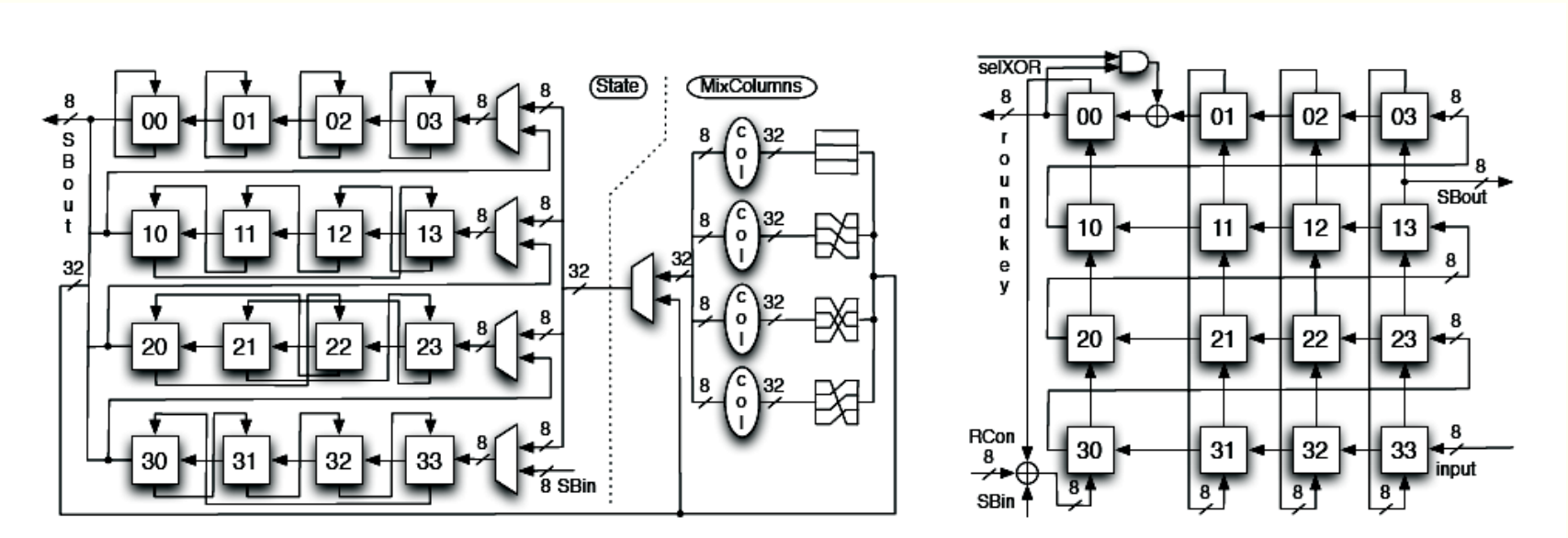
Moradi et al Eurocrypt 11



Circuit Description

- 16 banks of byte size registers ‘00’ to ‘33’ for the state.
- Similar arrangenment for the key.

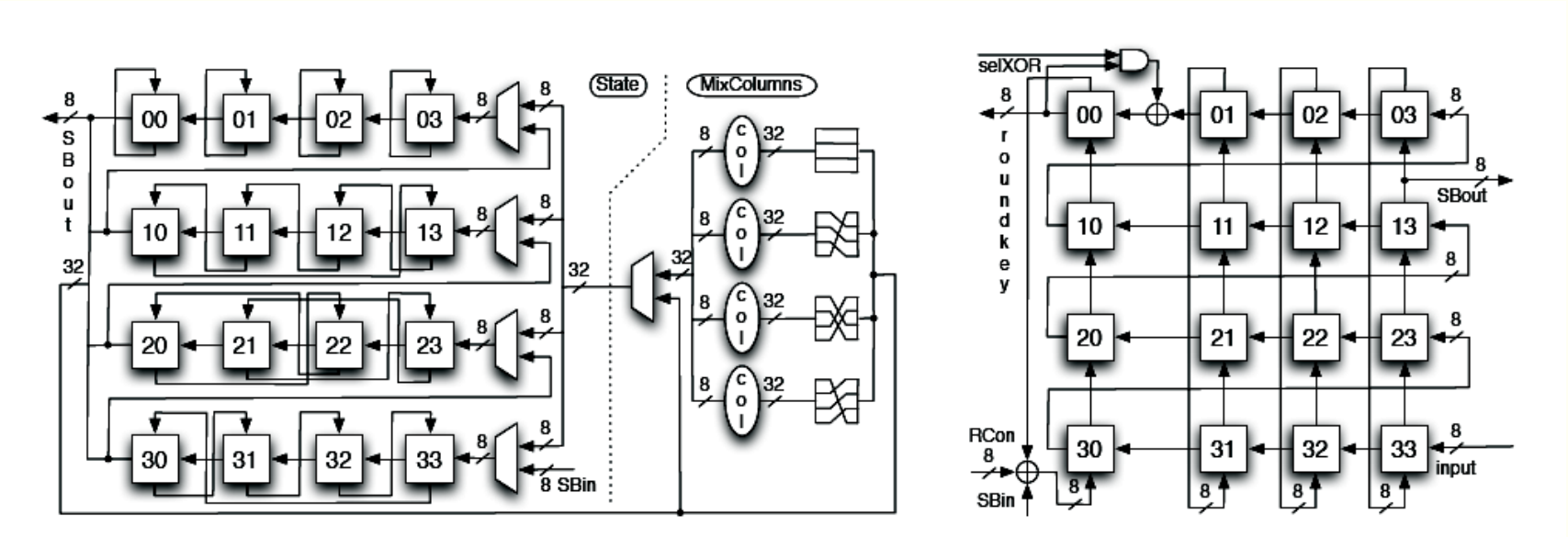
Moradi et al Eurocrypt 11



Circuit Description

- Each byte sized state register takes two inputs.
- One for serial loading and unloading, second for Shiftrow.

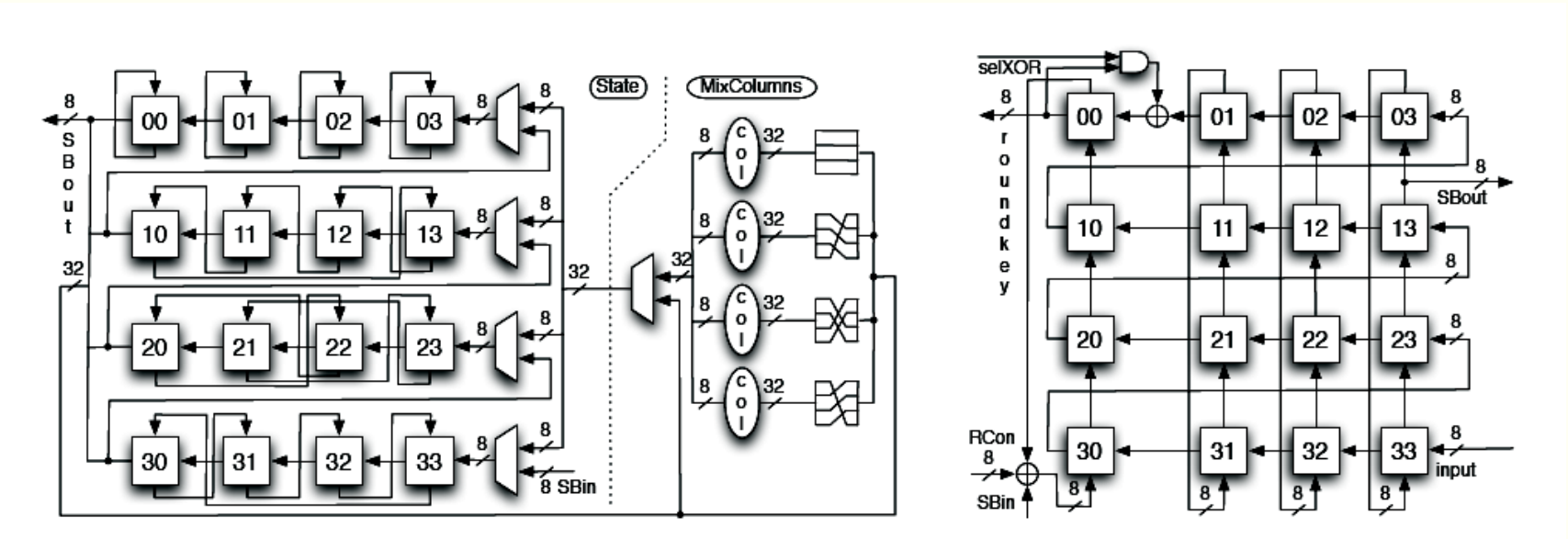
Moradi et al Eurocrypt 11



Circuit Description

- The connections in key register helps to do keyschedule.
- Two data movements: horizontal and vertical.

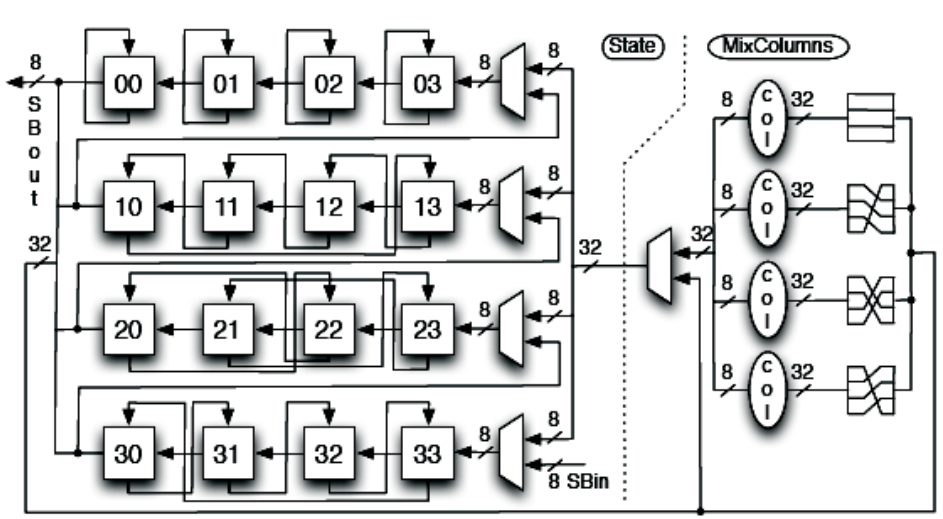
Moradi et al Eurocrypt 11

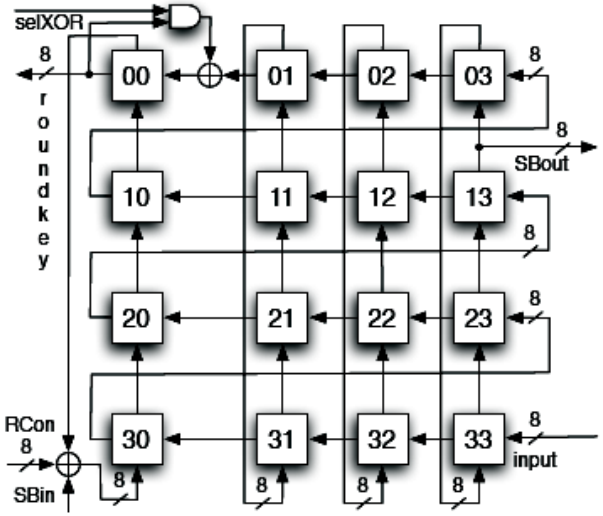


Circuit Description

- Scan flip-flops for each register: 6 GE.
- D Flip-flop + Mux takes 7.33 GE: save 1.33 GE per bit.

Moradi et al Eurocrypt 11

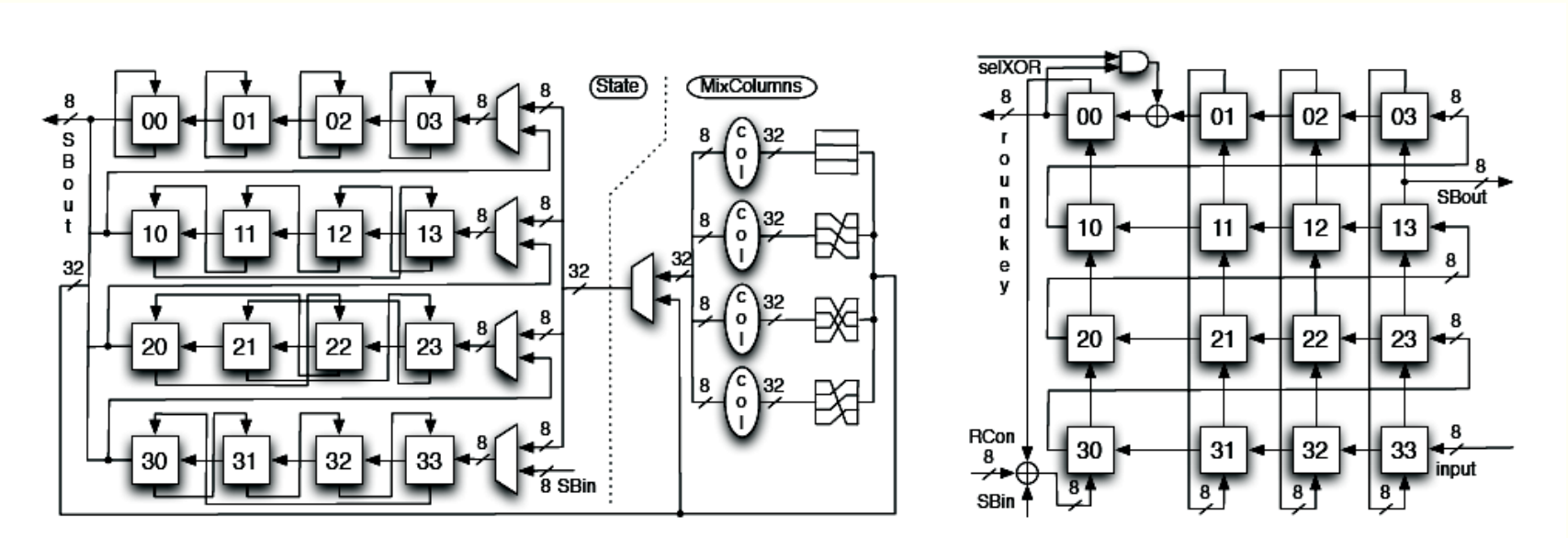




Circuit Description

- Only one S-box and one 8-bit xor for ARK (not shown).
- S-box uses Canright architecture.

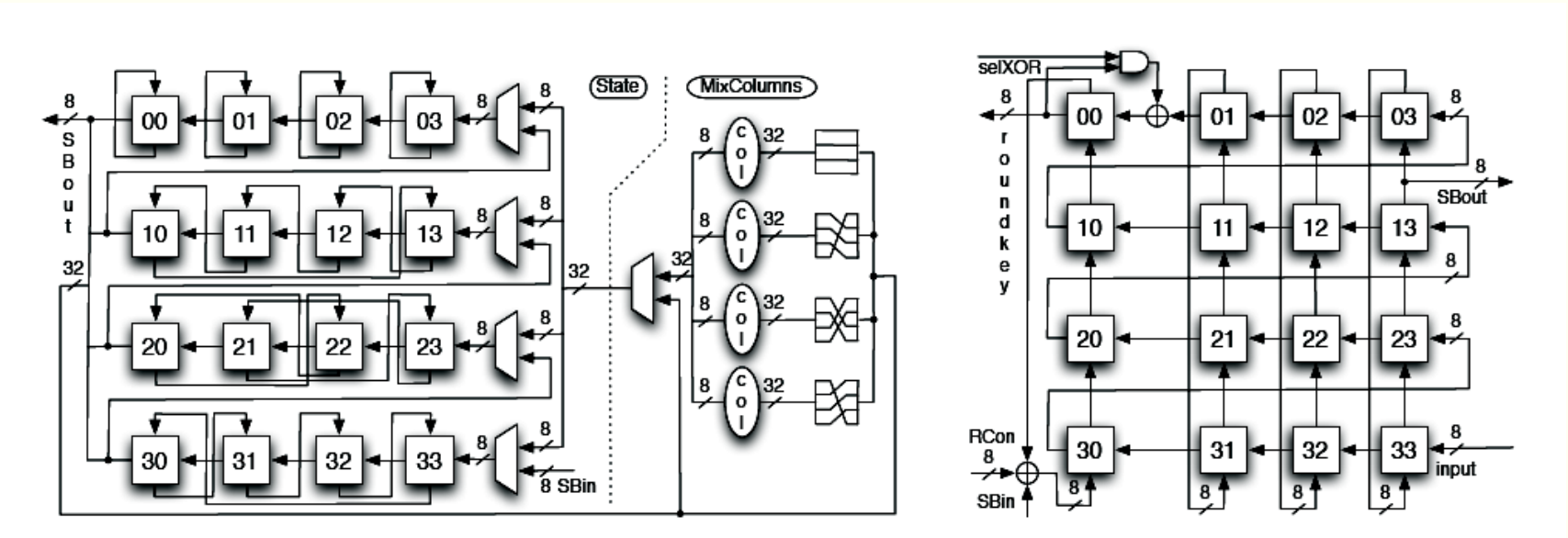
Moradi et al Eurocrypt 11



Circuit Description

- Mixcolumn implemented as logic block in $\{0,1\}^{32} \rightarrow \{0,1\}^{32}$.
- Takes 4 cycles to compute over the state.

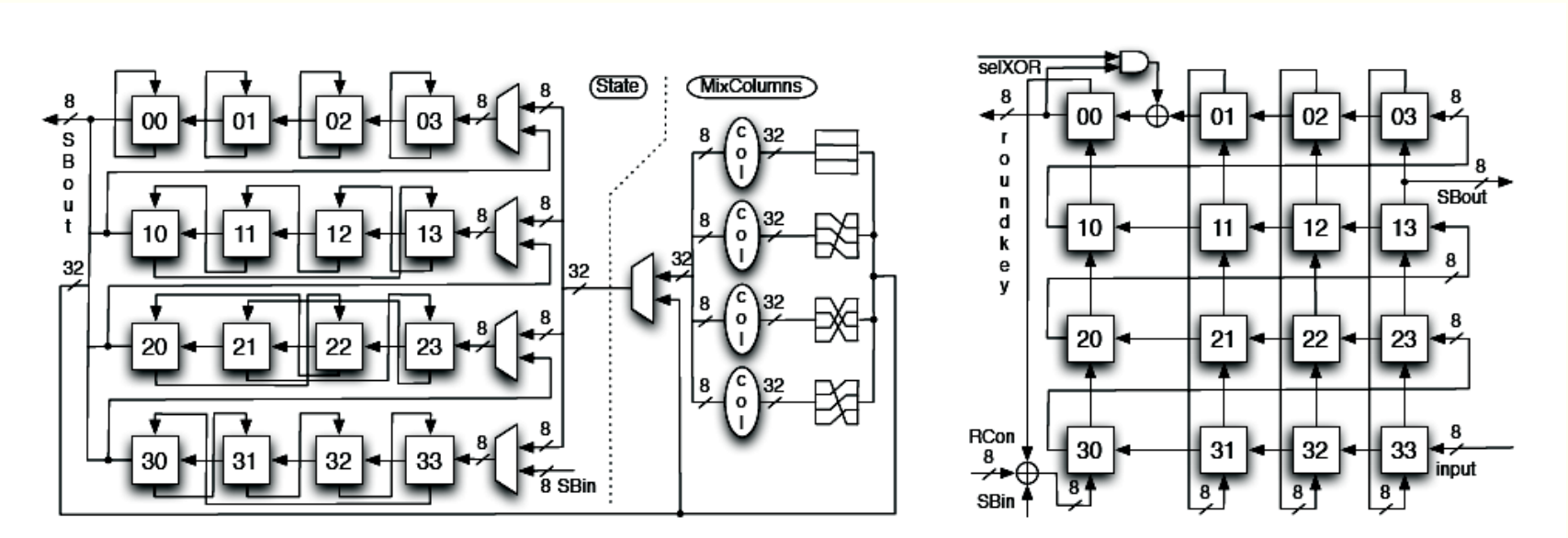
Moradi et al Eurocrypt 11



Circuit Description

- 32 bit Mux after Mixcolumn for 10th round bypass.
- 8 bit Mux before S-box to choose between state, key.

Moradi et al Eurocrypt 11

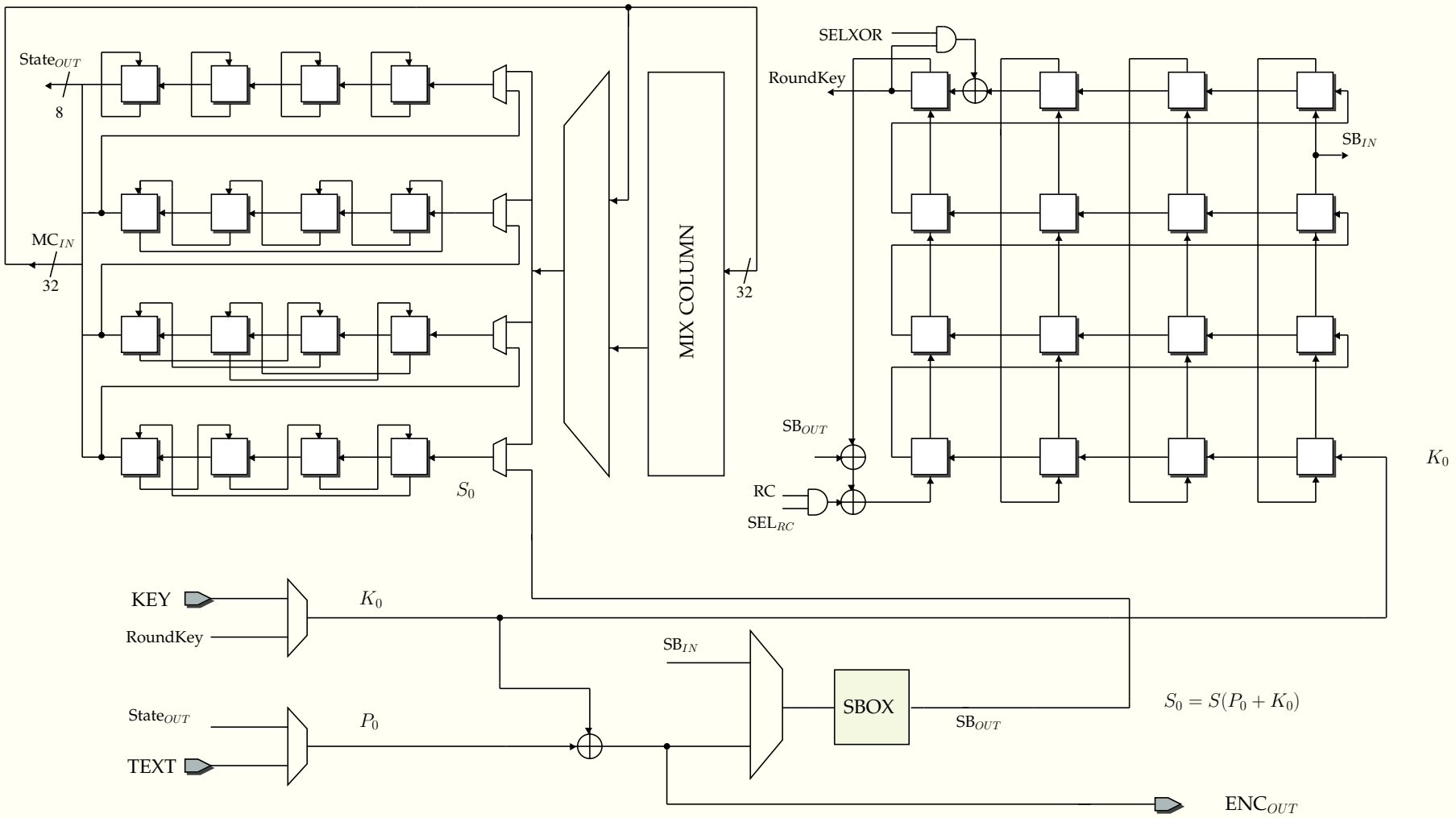


Circuit Description

- Round is computed in 21 cycles, encryption in 226 cycles.
- Special 21 cycle LFSR generates all control signals.

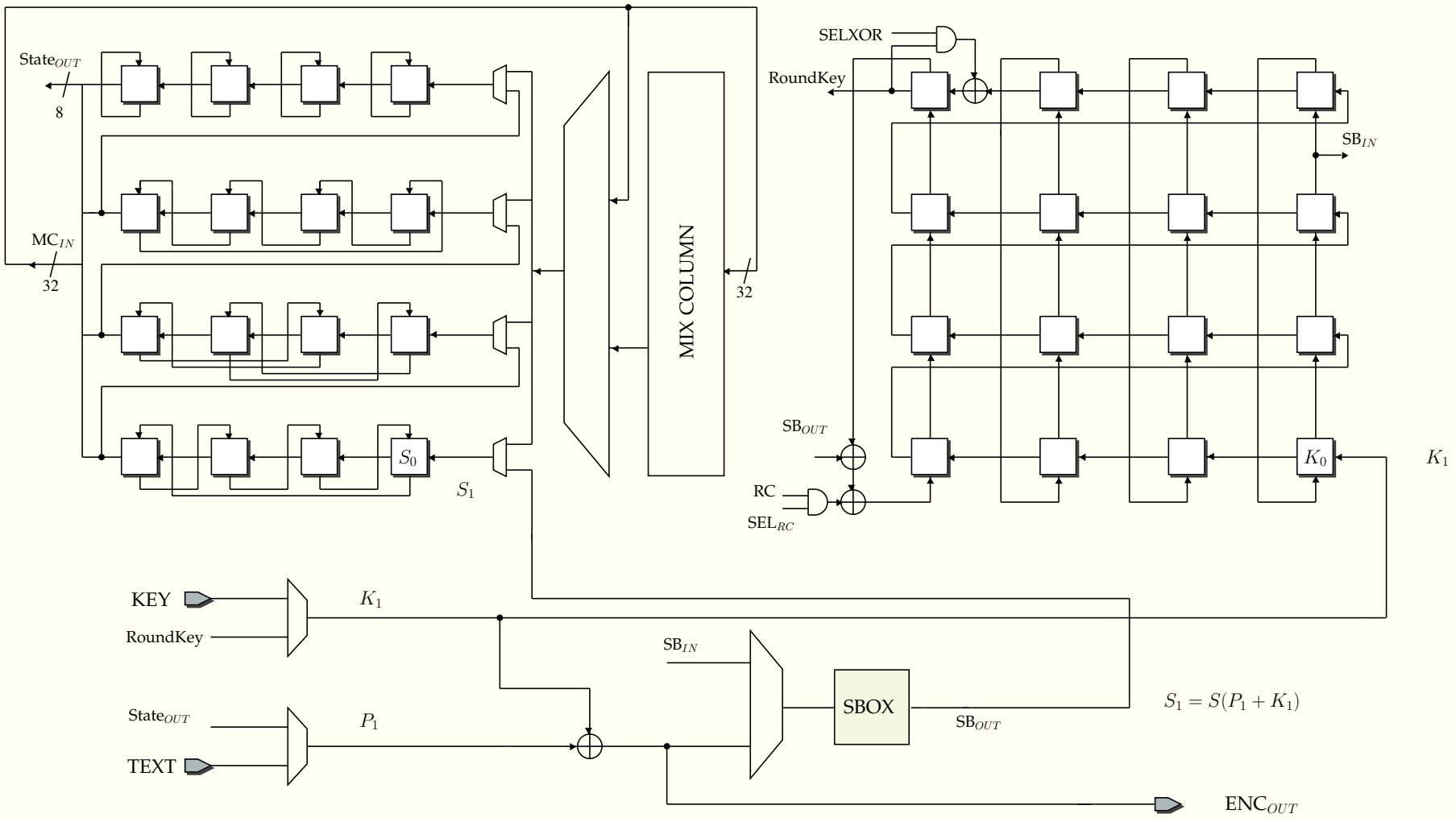
Data flow

Round 0, Cycle 5



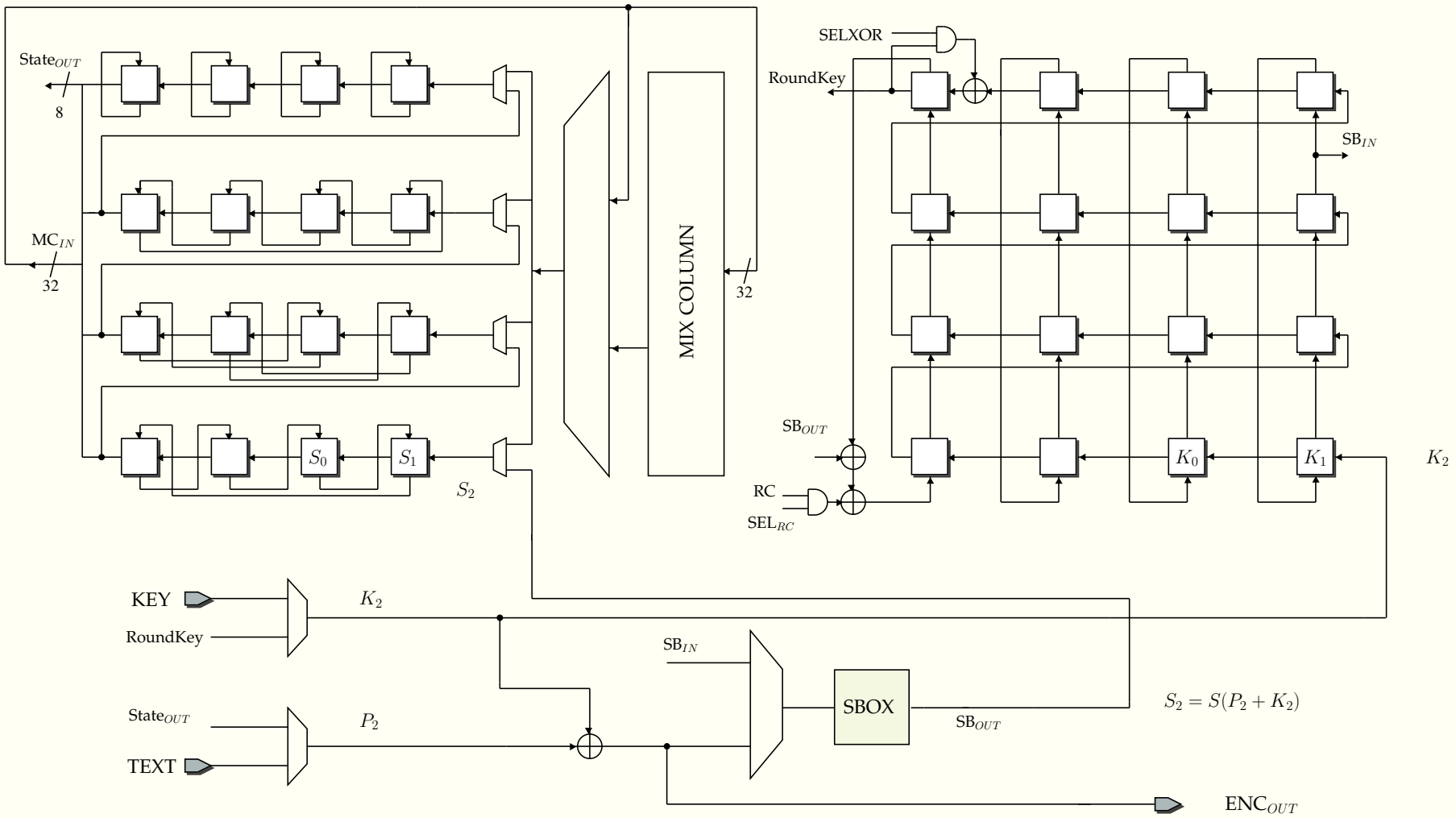
Data flow

Round 0, Cycle 6



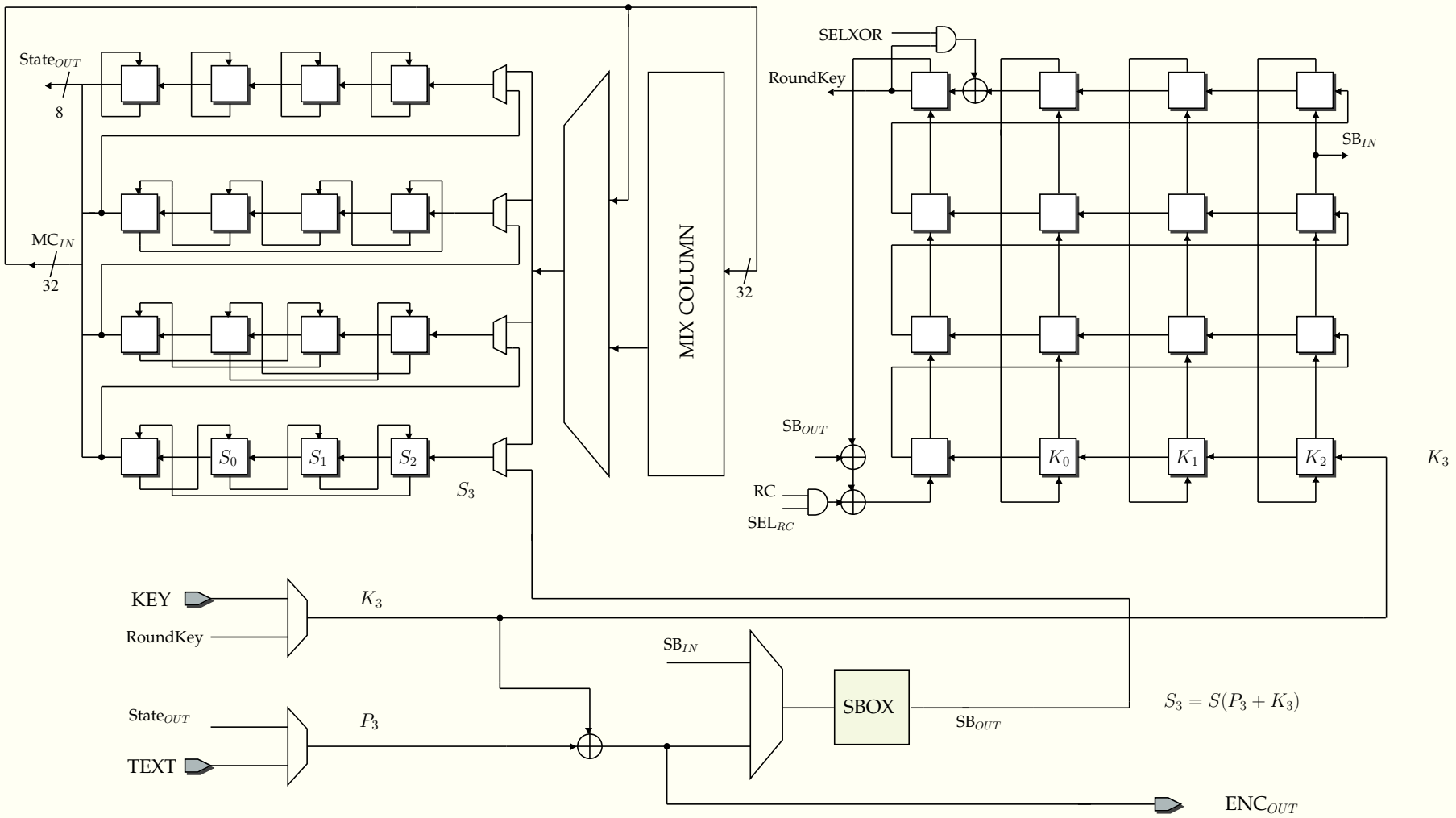
Data flow

Round 0, Cycle 7



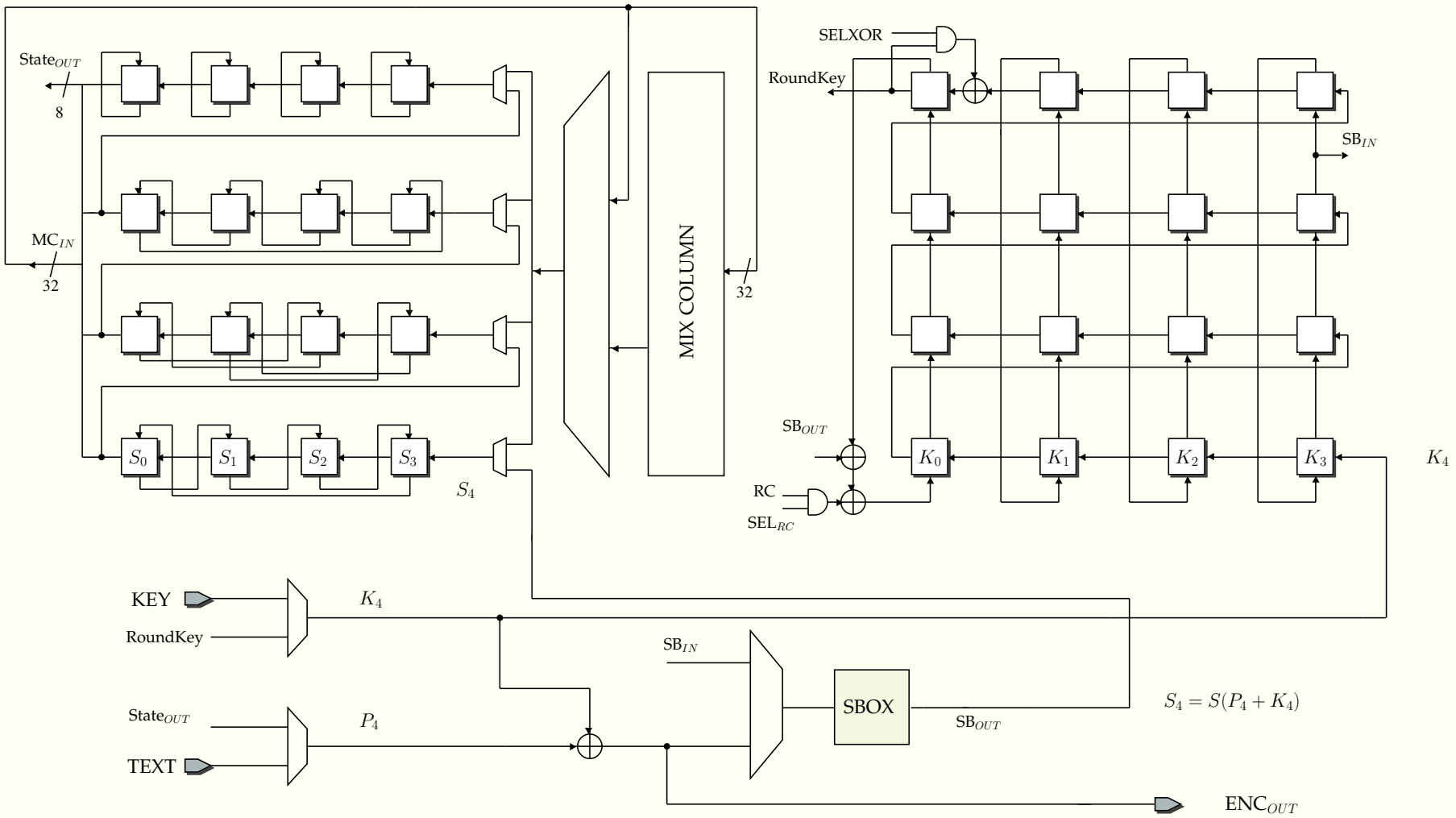
Data flow

Round 0, Cycle 8



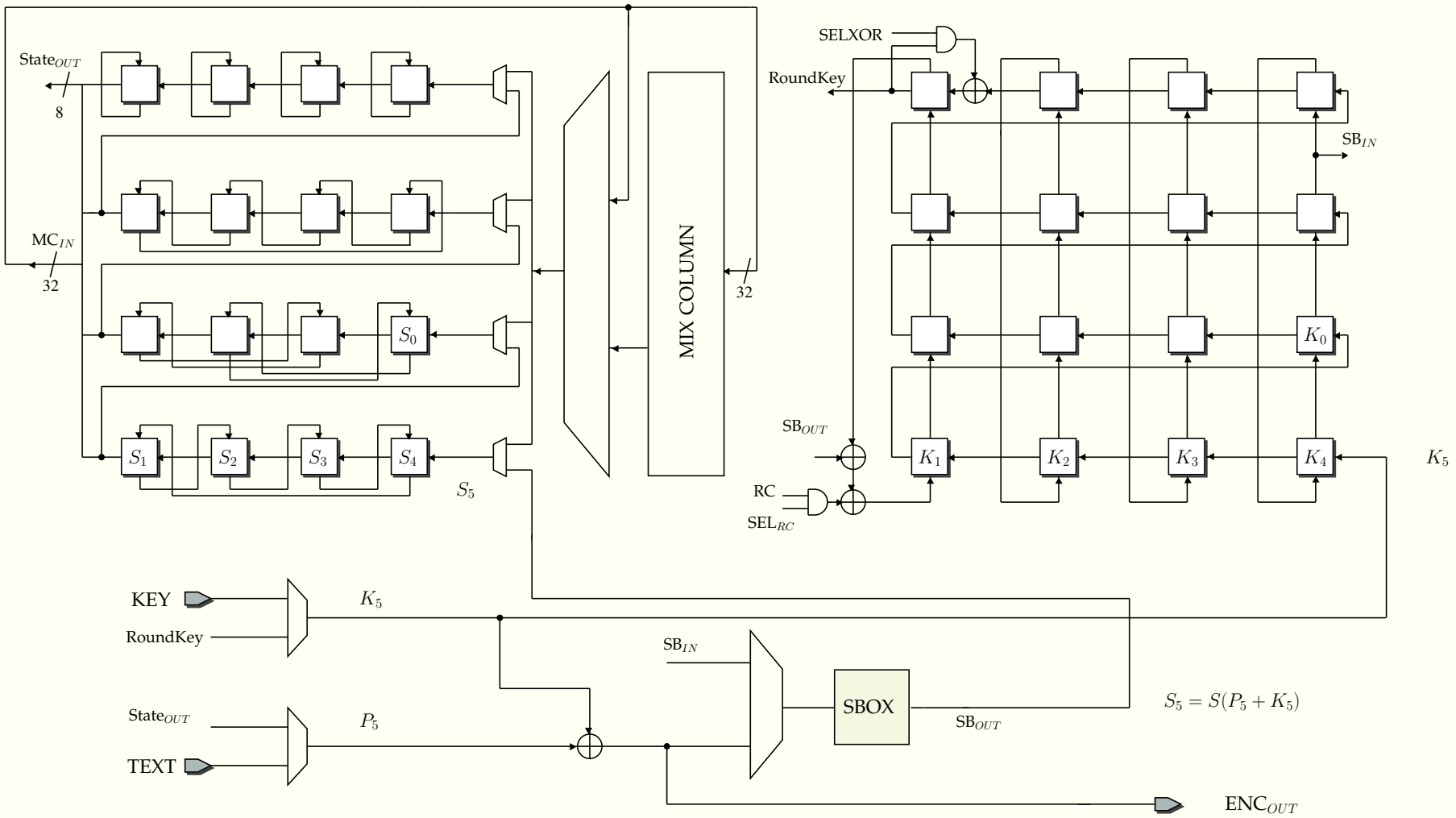
Data flow

Round 0, Cycle 9



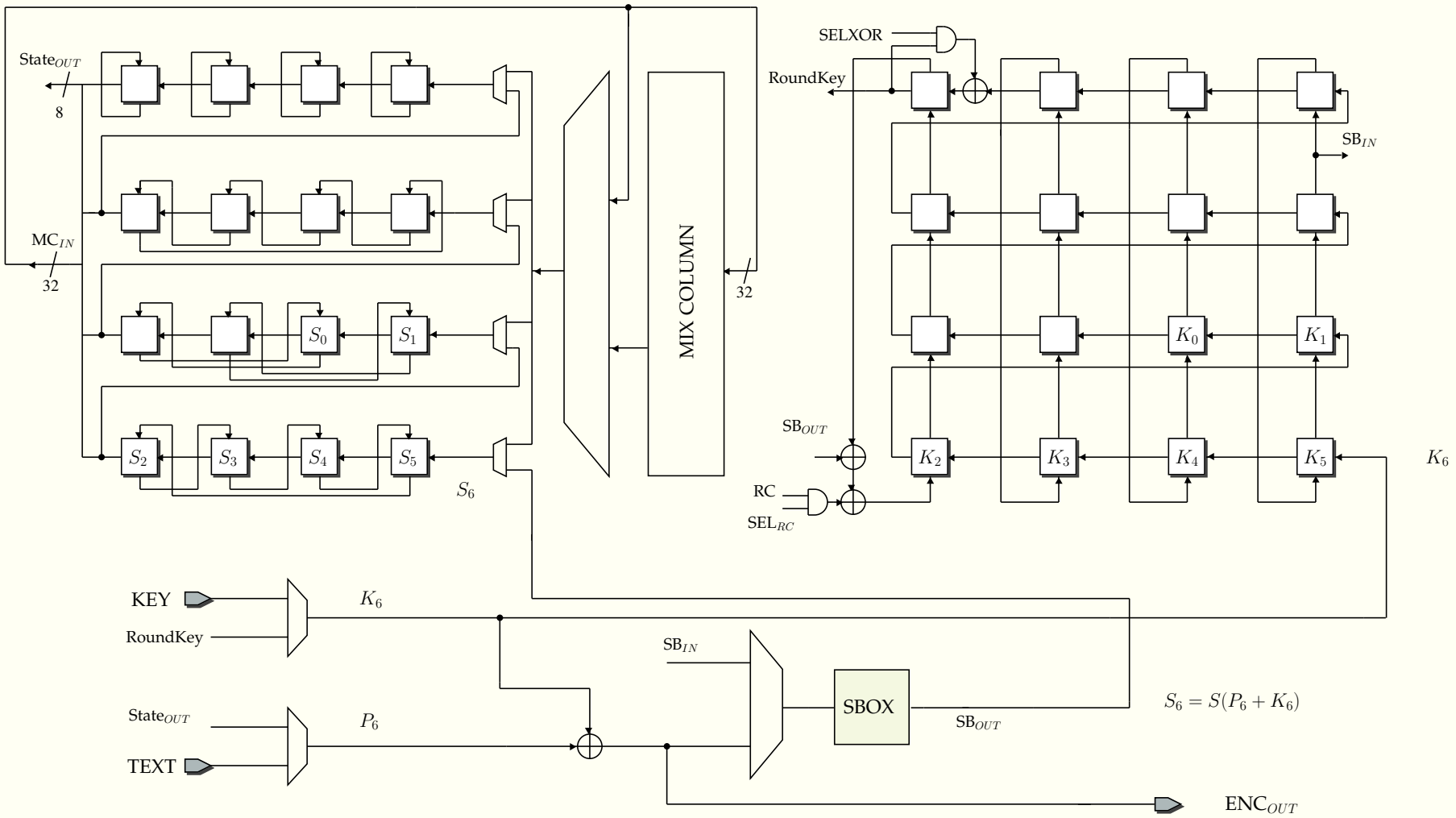
Data flow

Round 0, Cycle 10



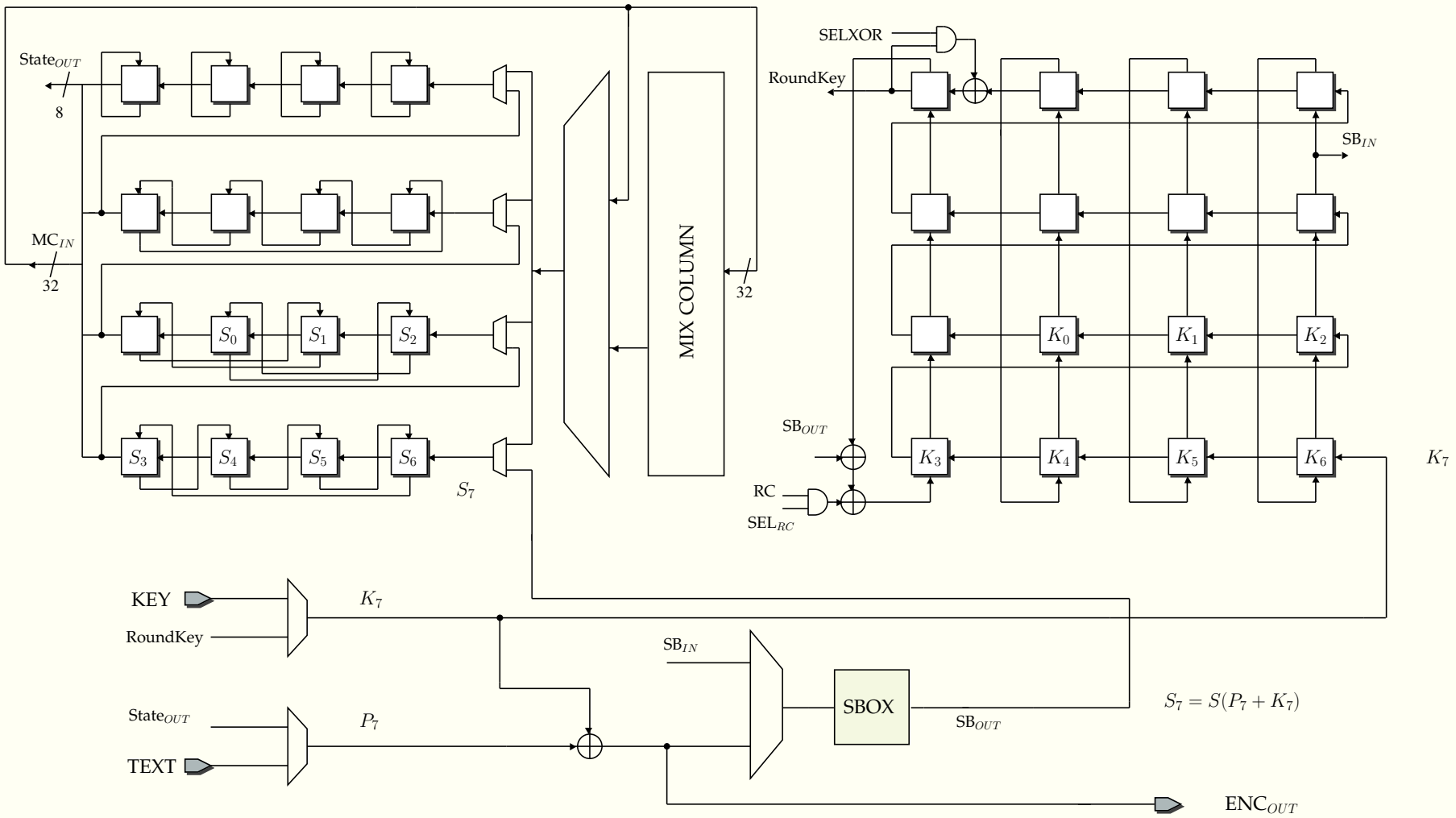
Data flow

Round 0, Cycle 11



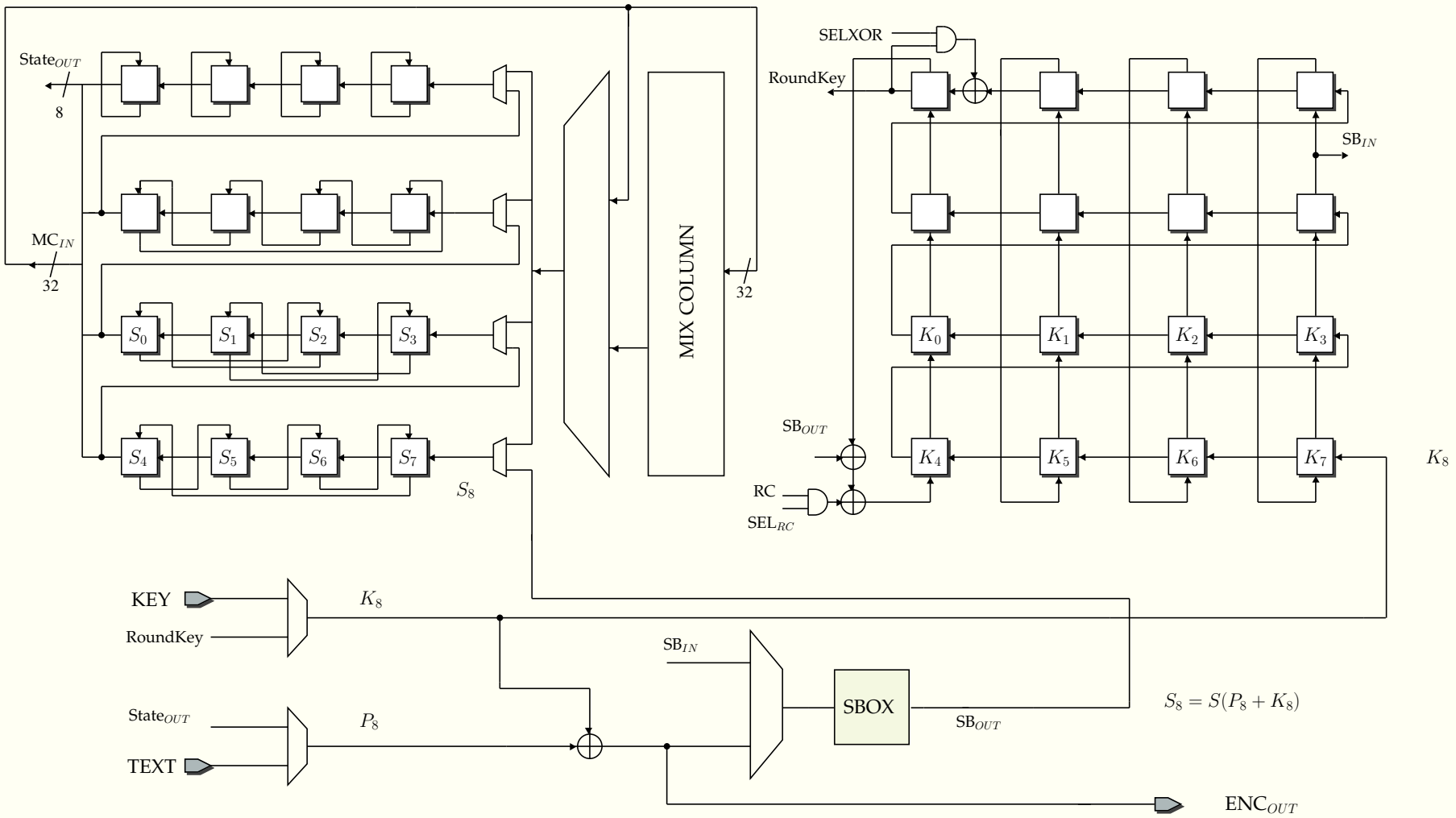
Data flow

Round 0, Cycle 12



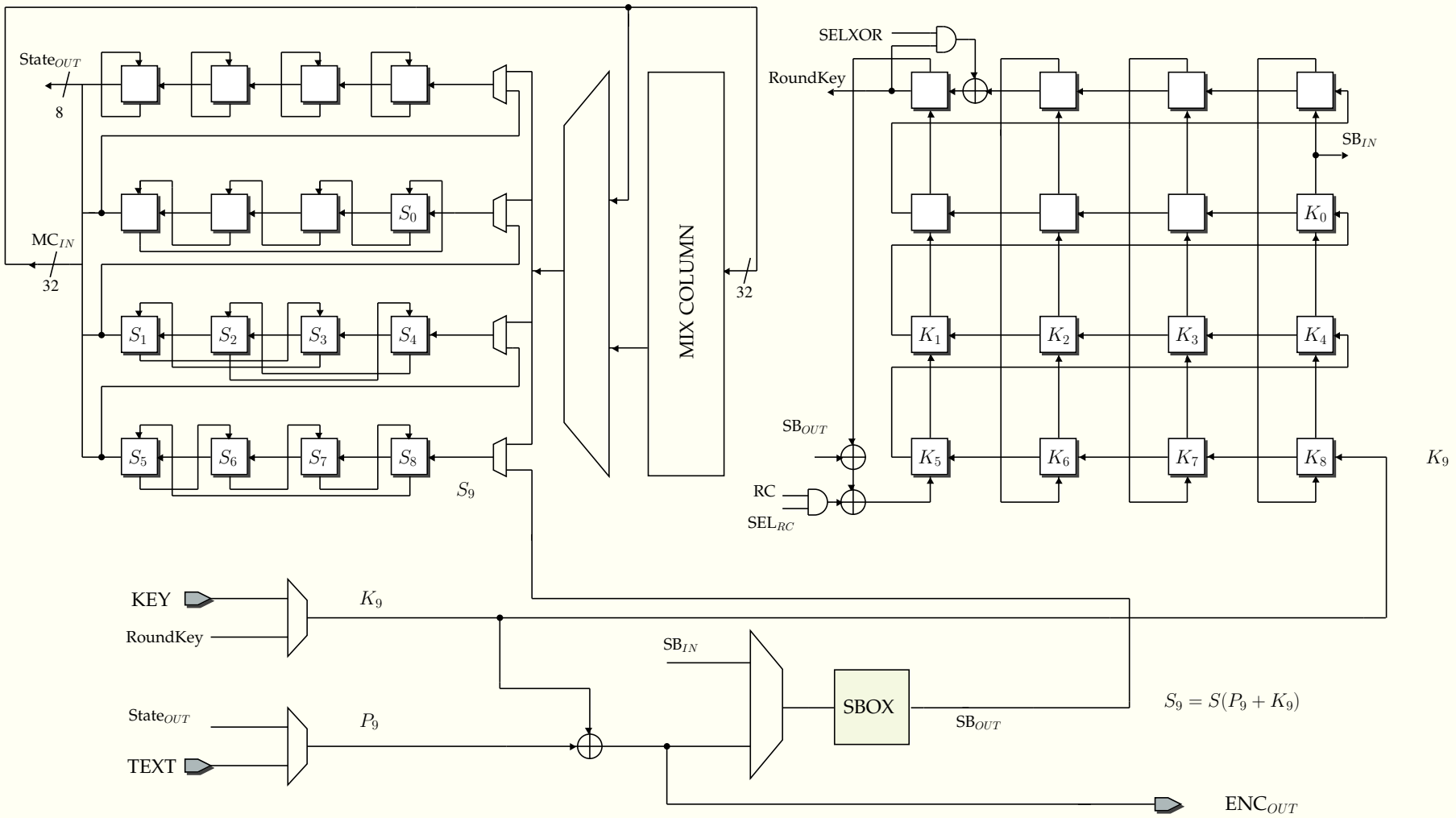
Data flow

Round 0, Cycle 13



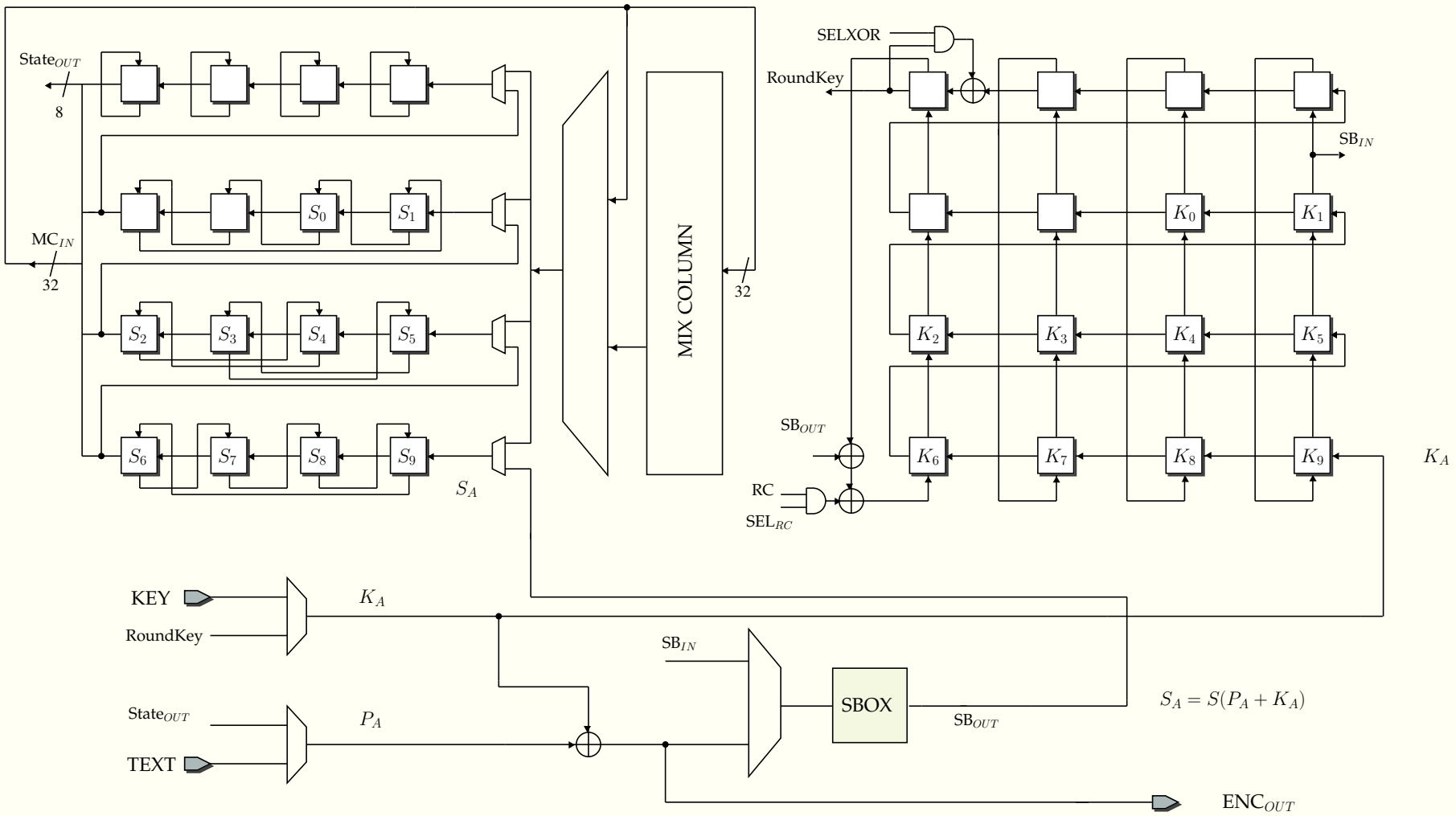
Data flow

Round 0, Cycle 14



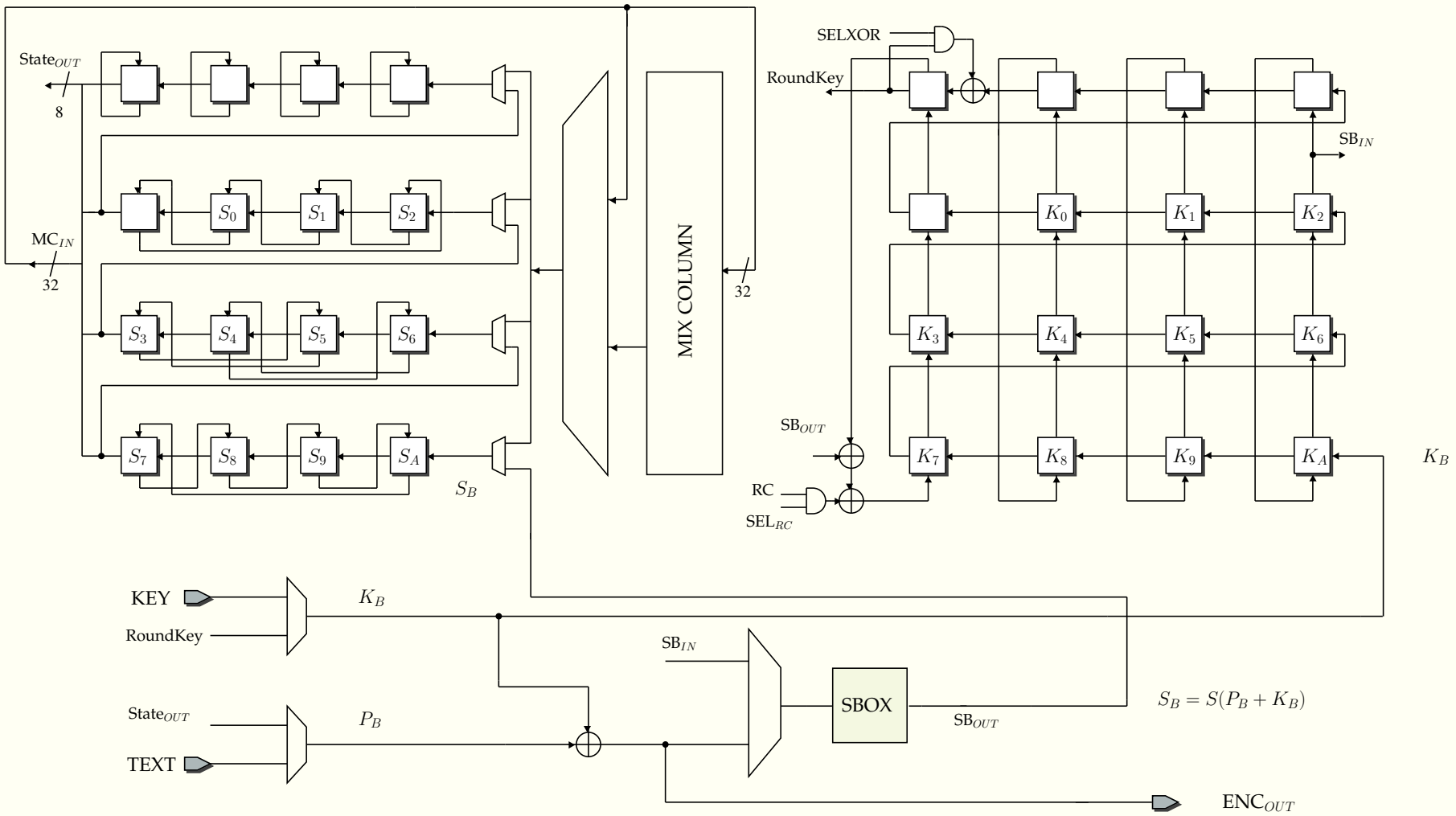
Data flow

Round 0, Cycle 15



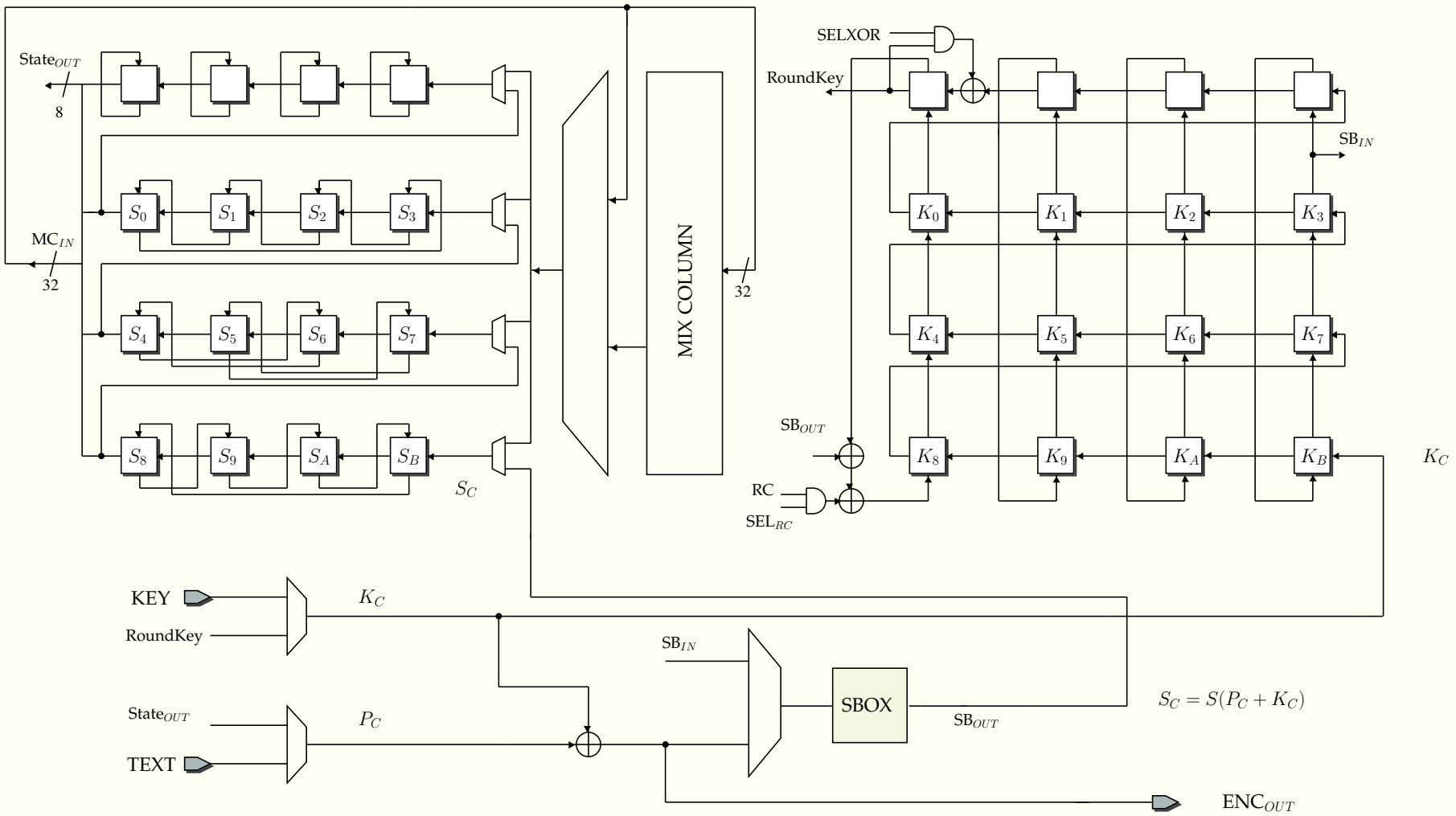
Data flow

Round 0, Cycle 16



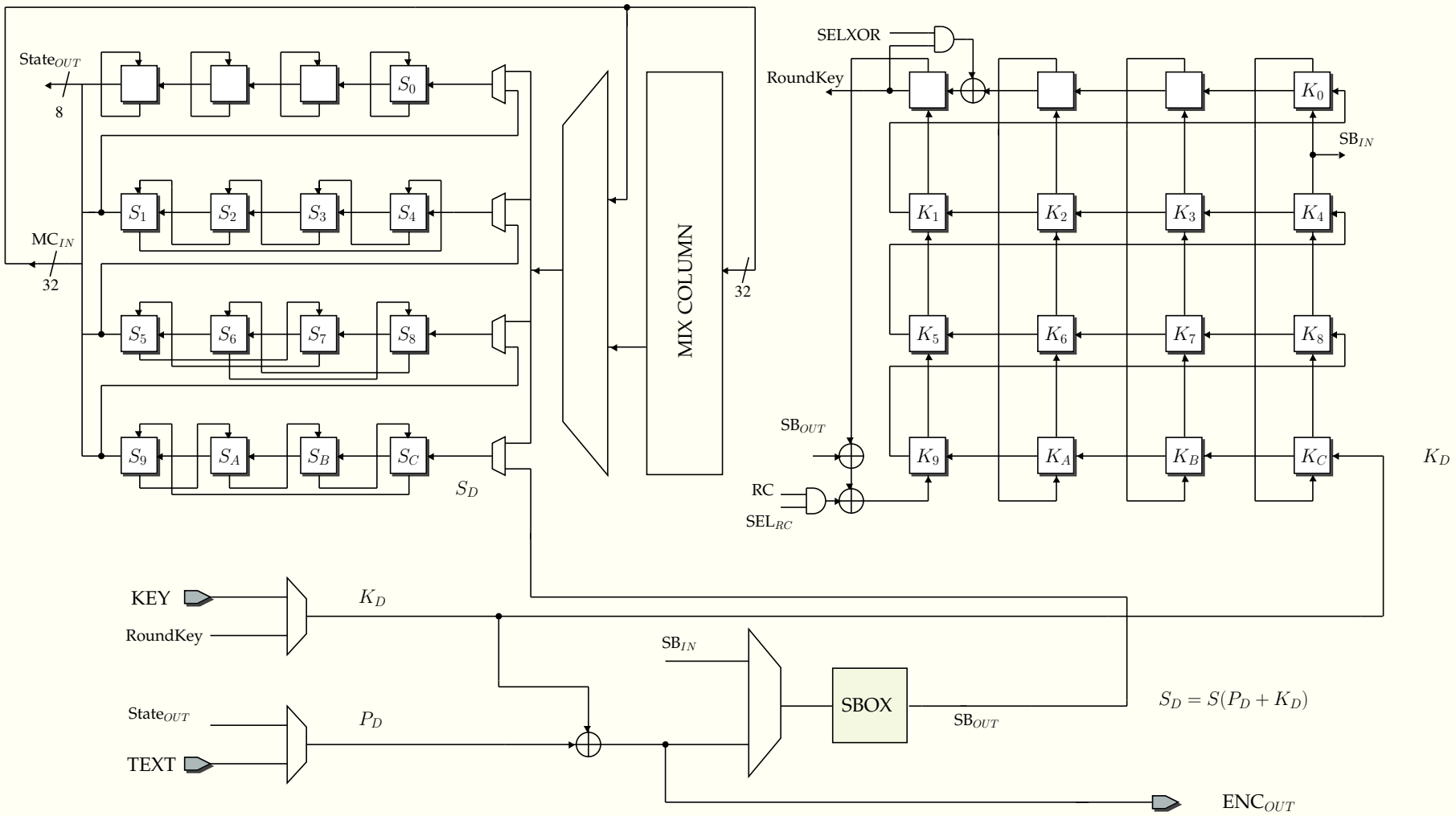
Data flow

Round 0, Cycle 17



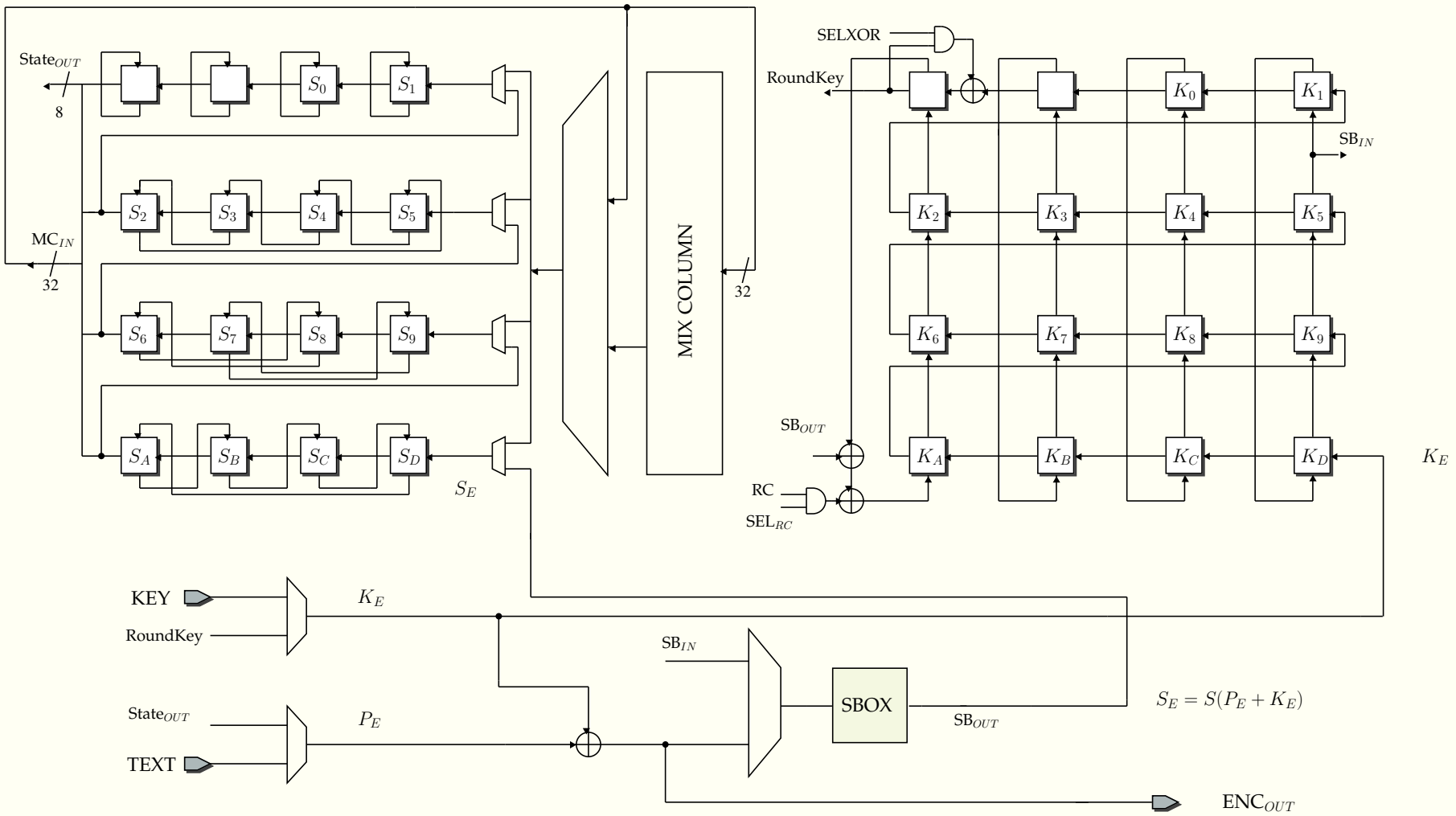
Data flow

Round 0, Cycle 18



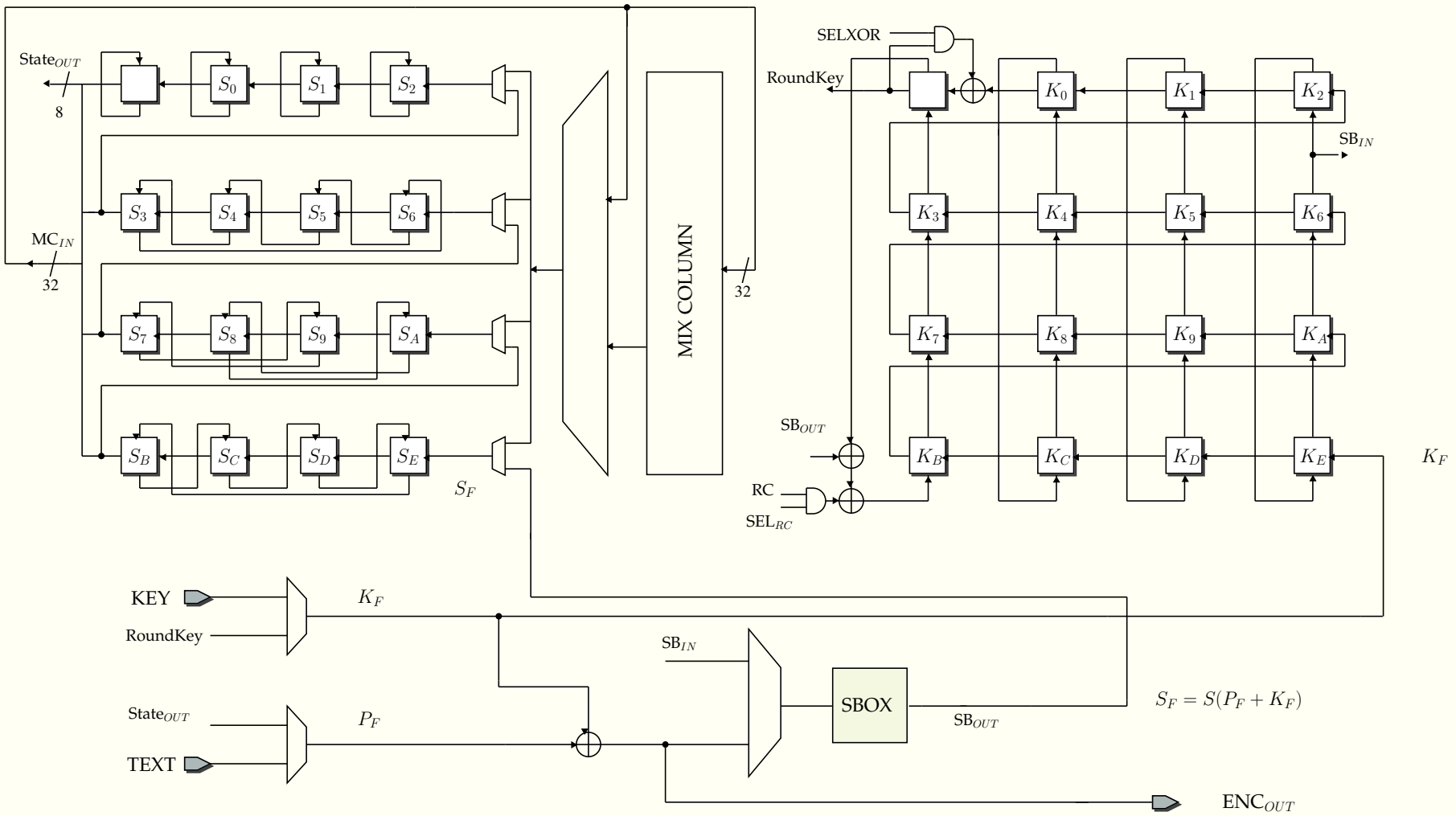
Data flow

Round 0, Cycle 19



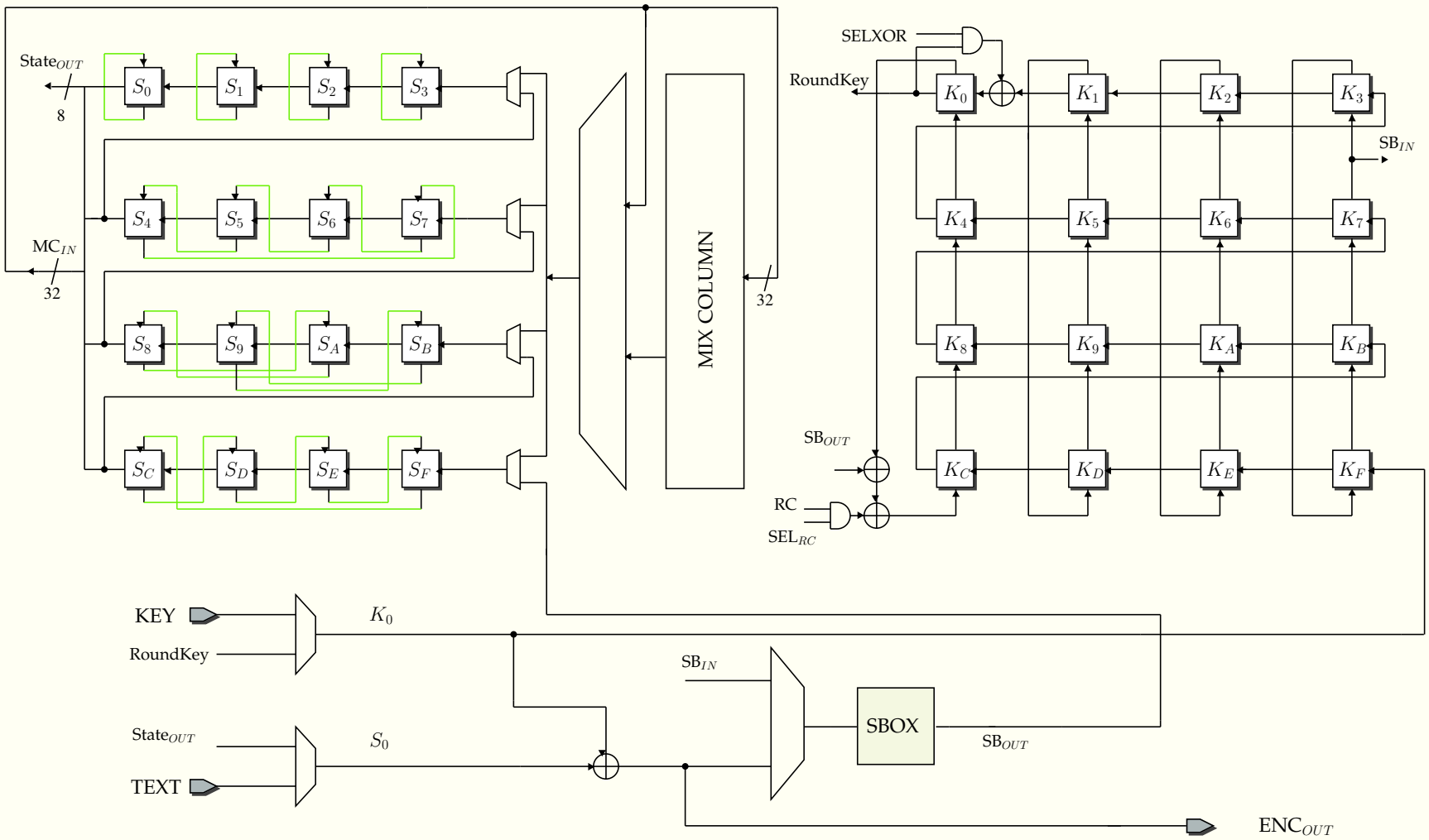
Data flow

Round 0, Cycle 20



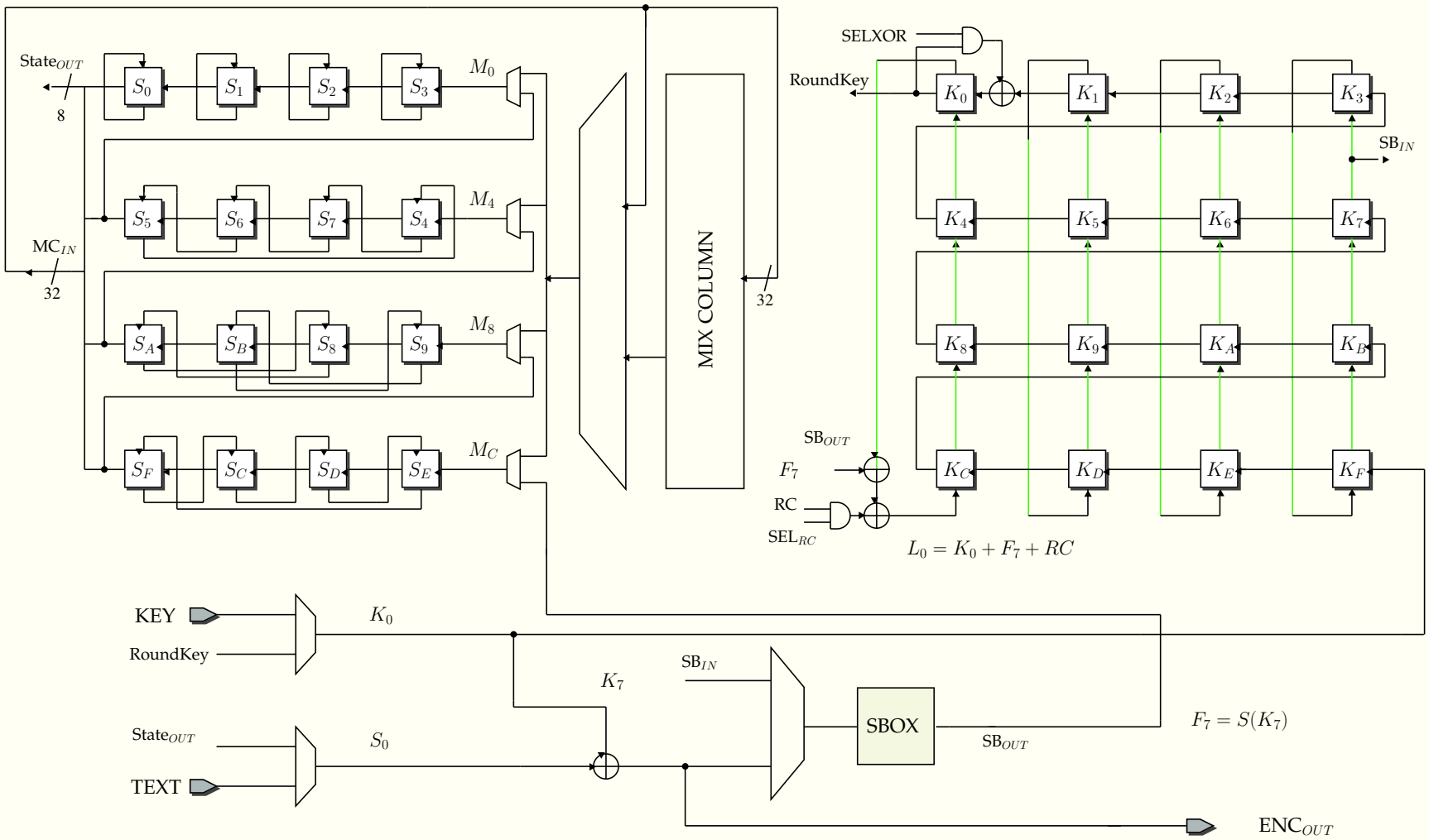
Data flow-SR+MC+ARK+SB

Round 1, Cycle 0



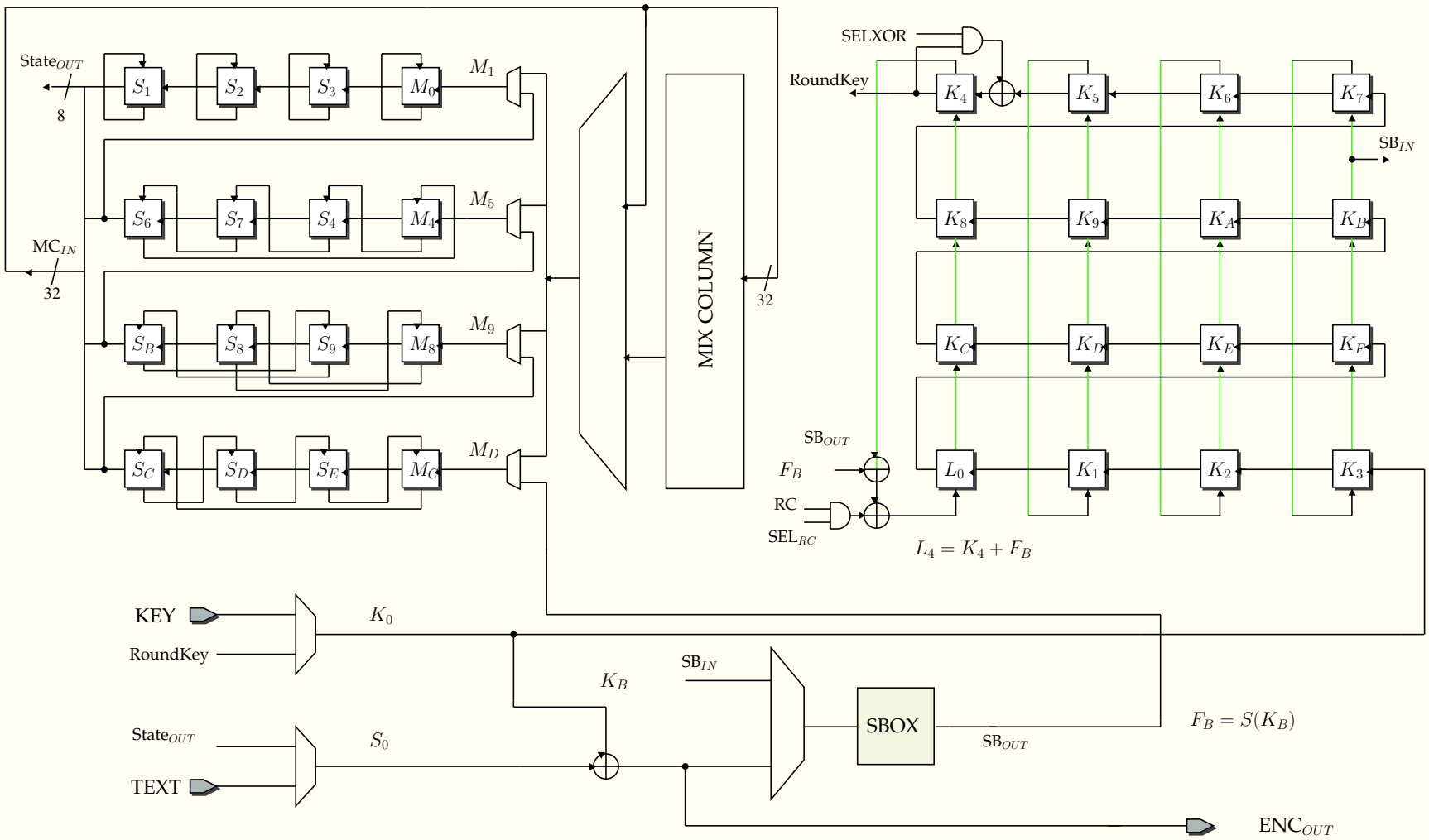
Data flow-SR+MC+ARK+SB

Round 1, Cycle 1



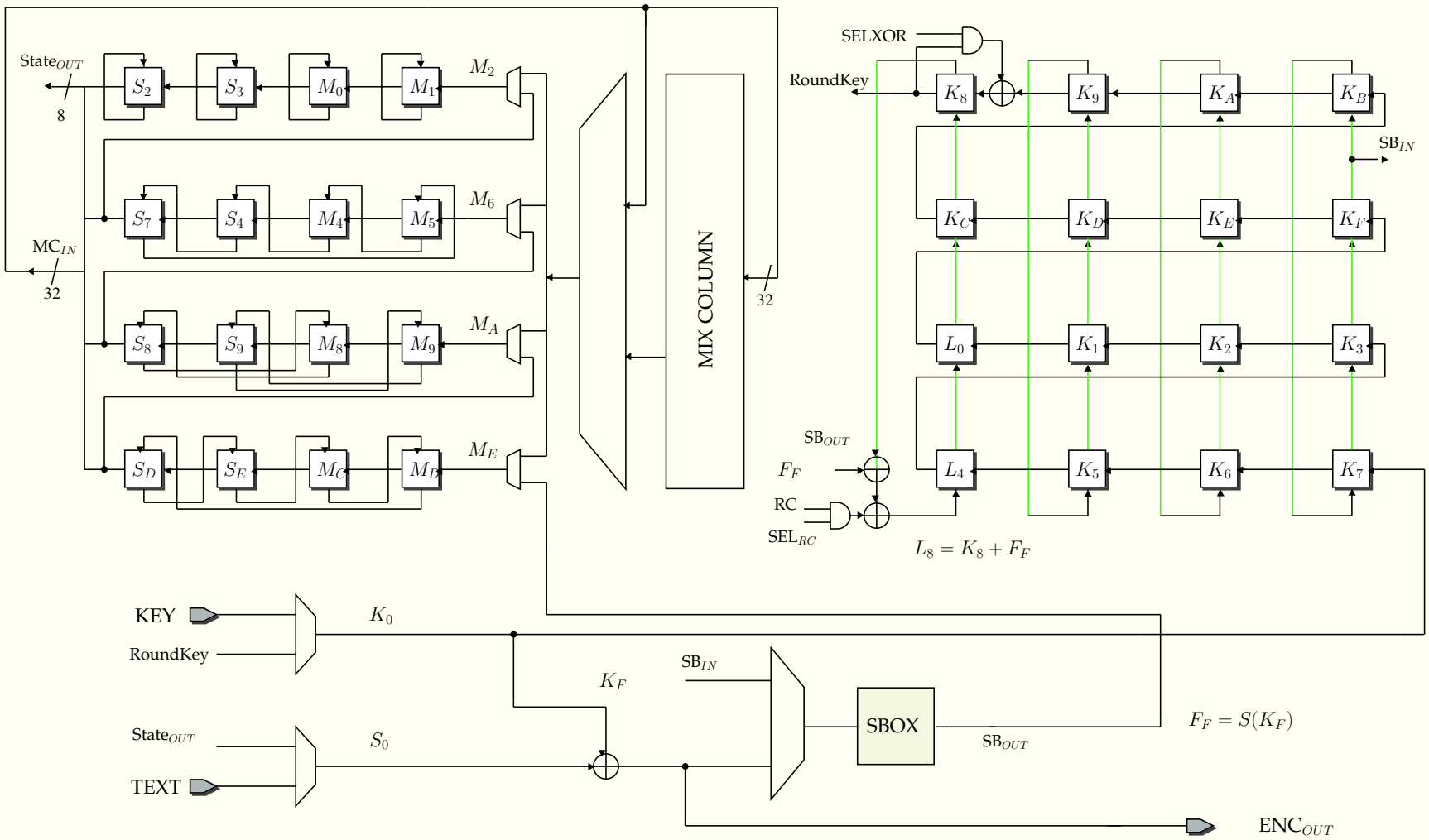
Data flow-SR+MC+ARK+SB

Round 1, Cycle 2



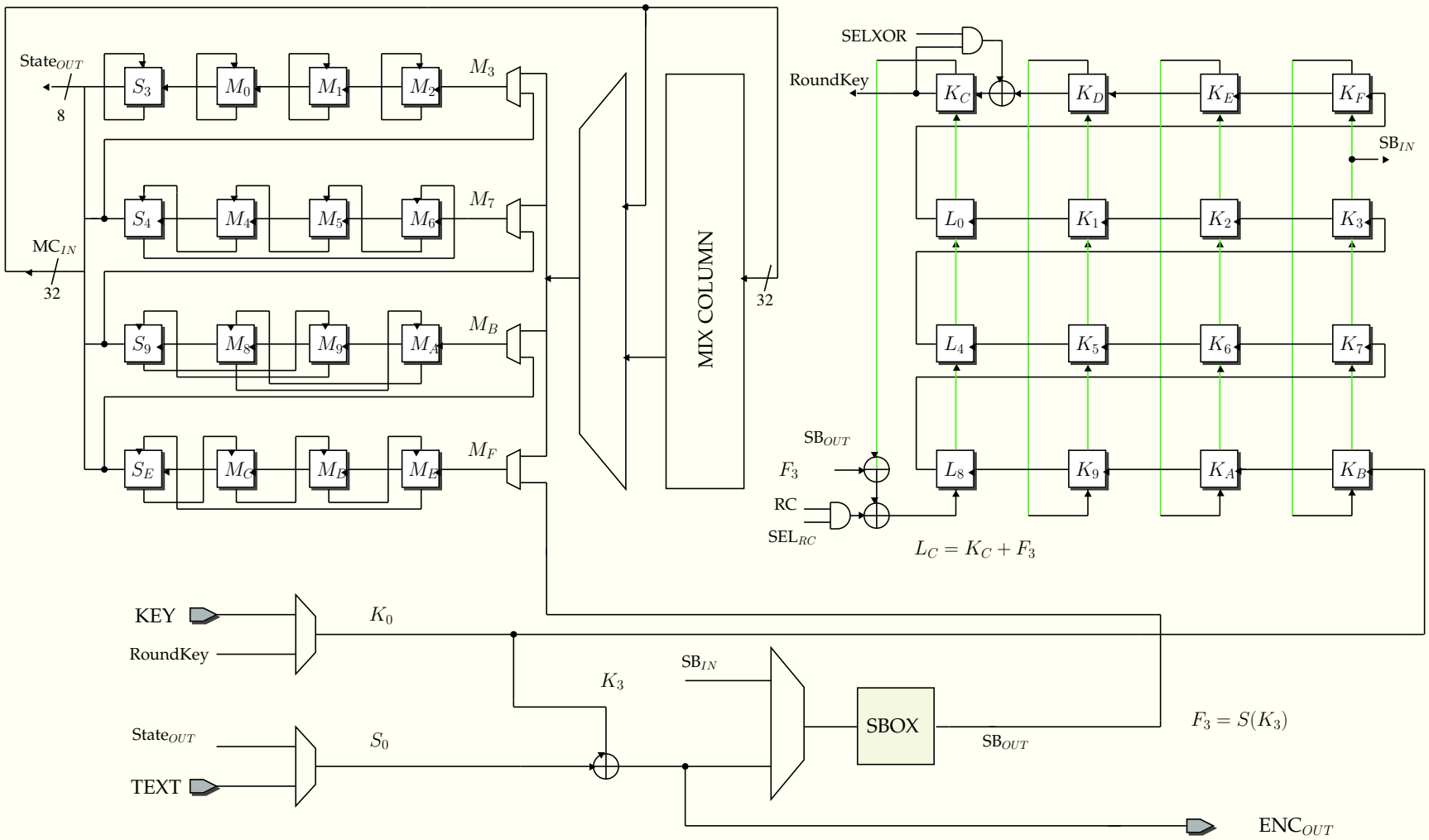
Data flow-SR+MC+ARK+SB

Round 1, Cycle 3



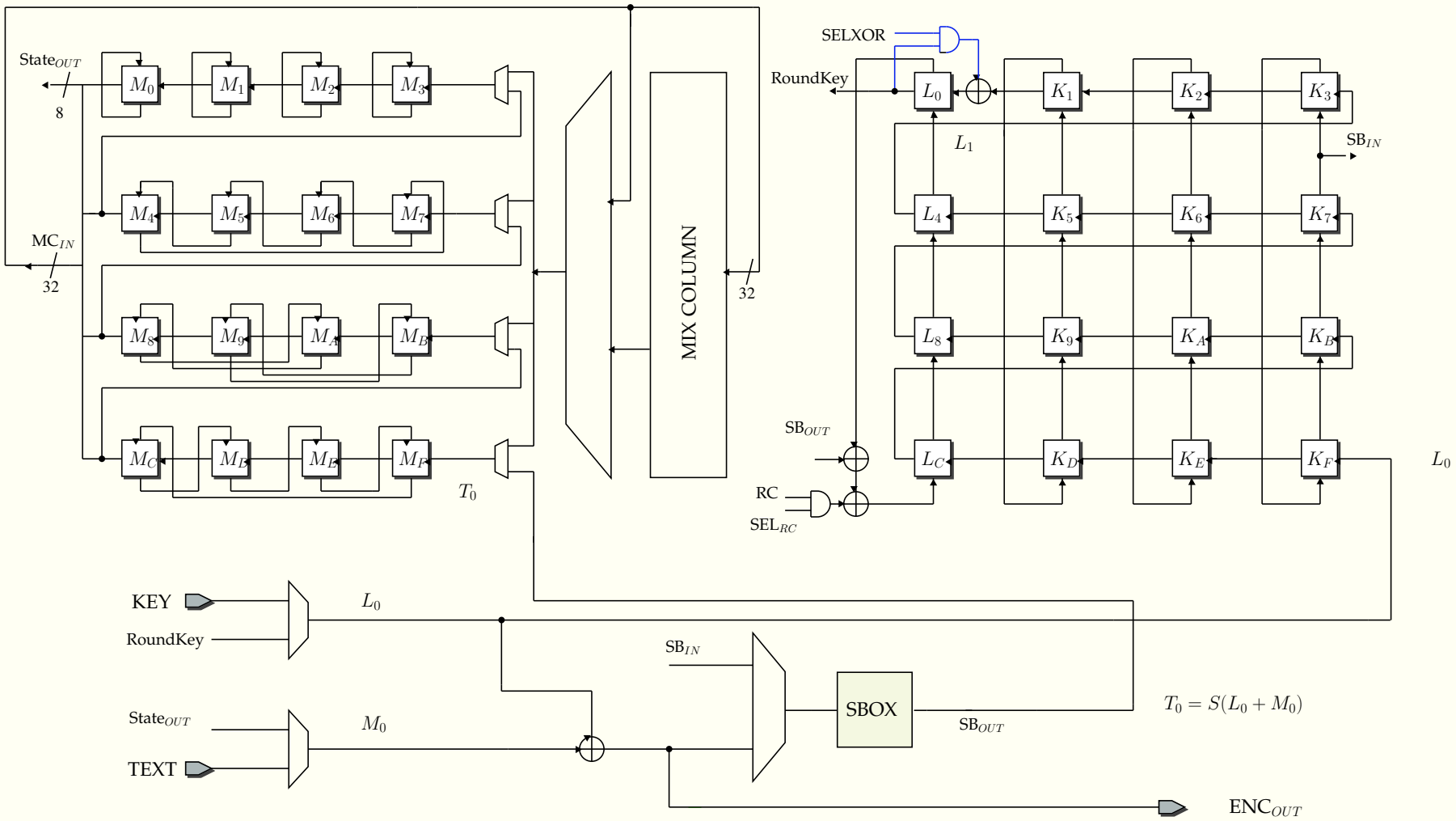
Data flow-SR+MC+ARK+SB

Round 1, Cycle 4



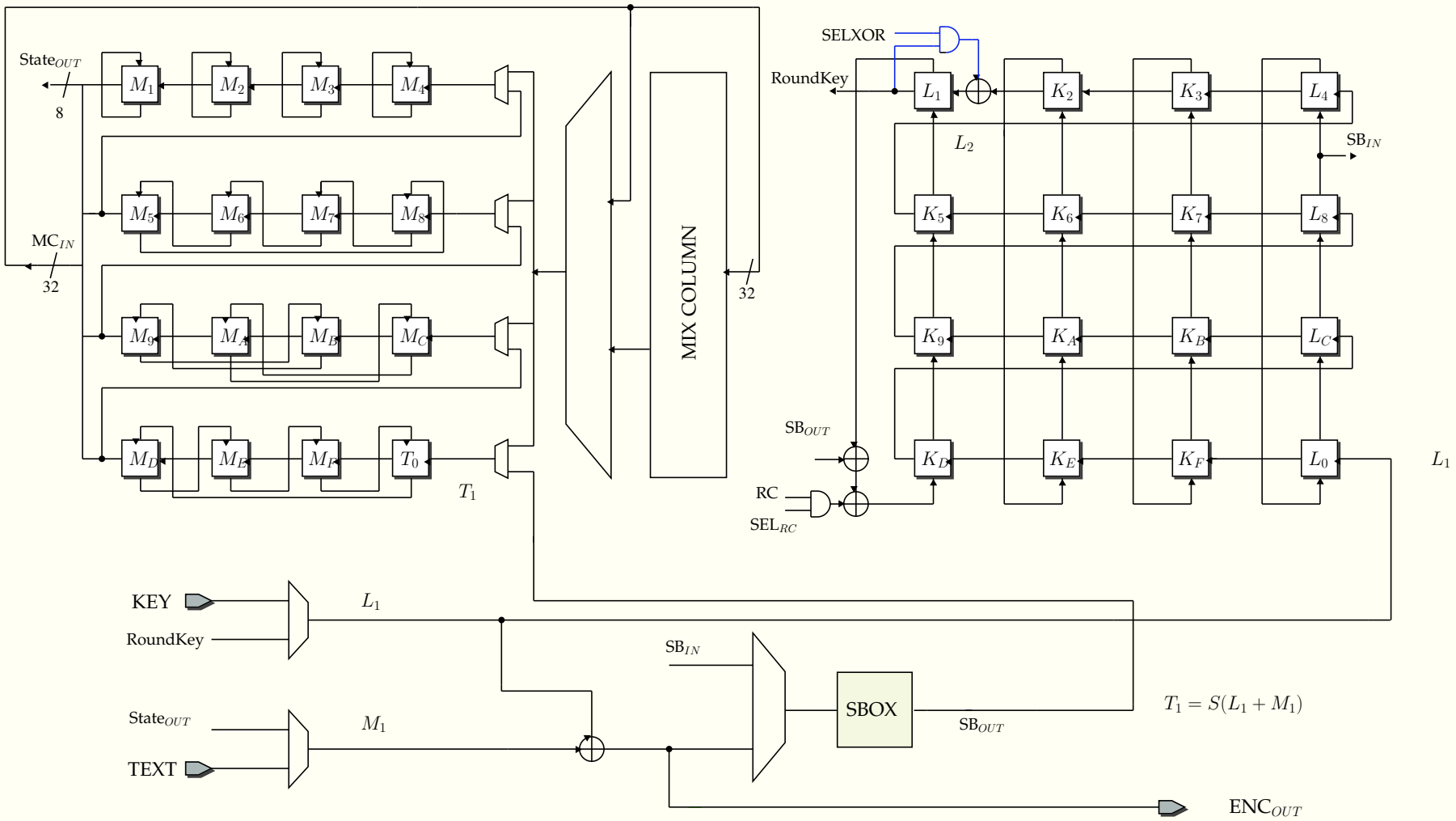
Data flow-SR+MC+ARK+SB

Round 1, Cycle 5



Data flow-SR+MC+ARK+SB

Round 1, Cycle 6



Atomic AES

Principal Issues: Inverse Shiftrow

- Implement Shiftrow and Inverse Shiftrow in same setup.
  Potentially one extra Mux for each 8-bit register.
- Can we do better?

Observation 1

For the 0^{th} and the 2^{nd} rows of the AES state, Shiftrow and Inverse Shiftrow bring about the same transformation.

$\Rightarrow$ **No change of logic required in the 0th and 2nd rows !!**

Principal Issues: Inverse Shiftrow

- Implement Shiftrow and Inverse Shiftrow in same setup.

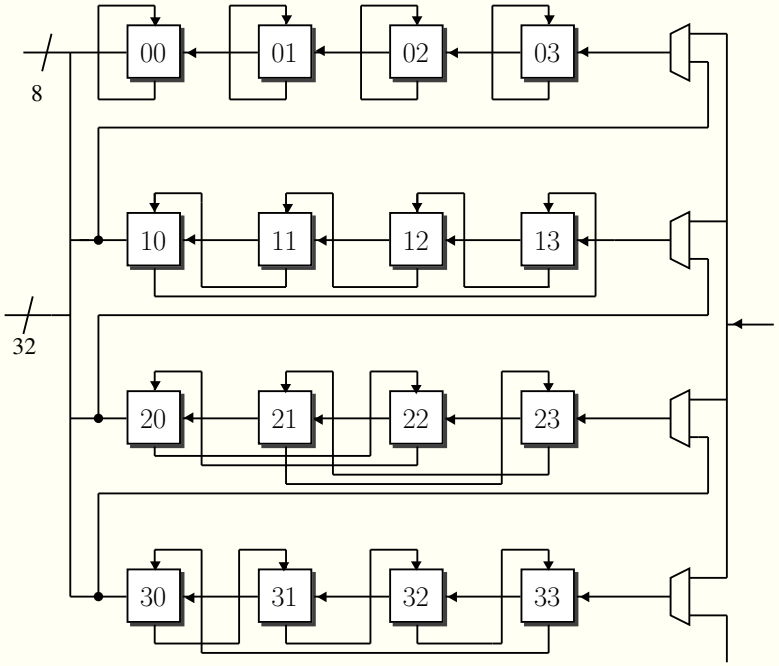
  Potentially one extra Mux for each 8-bit register.

- Can we do better?

Observation 2

For the 1^{st} and the 3^{rd} rows of the AES state, Shiftrow and Inverse Shiftrow bring about opposite transformations. Which is to say, that the Shiftrow operation on the 1^{st} row brings about the same transformation as the Inverse Shiftrow on the 3^{rd} row and vice versa.

Principal Issues: Inverse Shiftrow

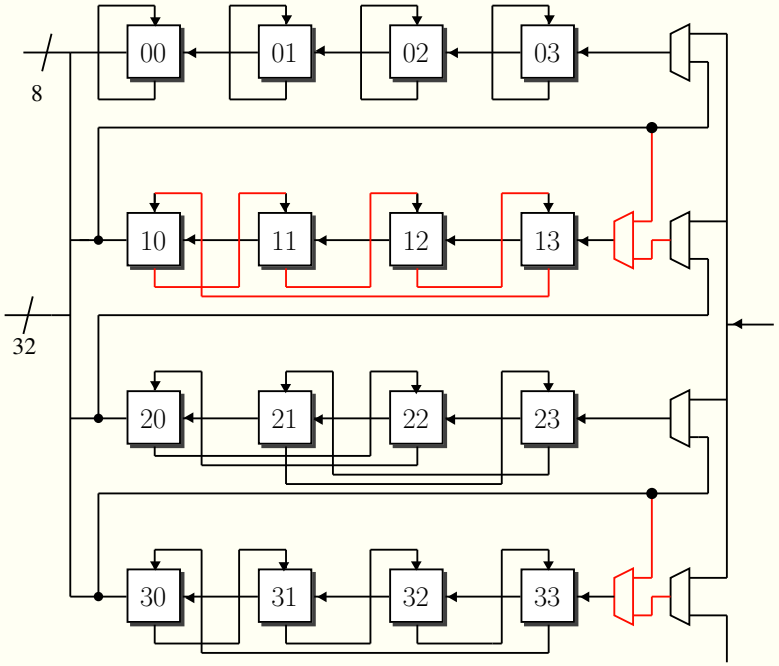


- Each register has 2 connections
  1. Serial loading/unloading 2. Shiftrows
- For regs 10,11,12 both connections are same.

  Rewire second connection for Shiftrow^{-1}. Extra mux required for 13.

  No changes in third row except extra mux required for 33 !! Why ?

Principal Issues: Inverse Shiftrow



- Each register has 2 connections
  1. Serial loading/unloading 2. Shiftrows
- For regs 10,11,12 both connections are same.

  Rewire second connection for Shiftrow^{-1}. Extra mux required for 13.

  No changes in third row except extra mux required for 33 !! Why ?

Principal Issues: Inv. Keyschedule

- $K_0, K_1, K_2, K_3 \rightarrow$ Current roundkey column
- $L_0, L_1, L_2, L_3 \rightarrow$ Next roundkey column

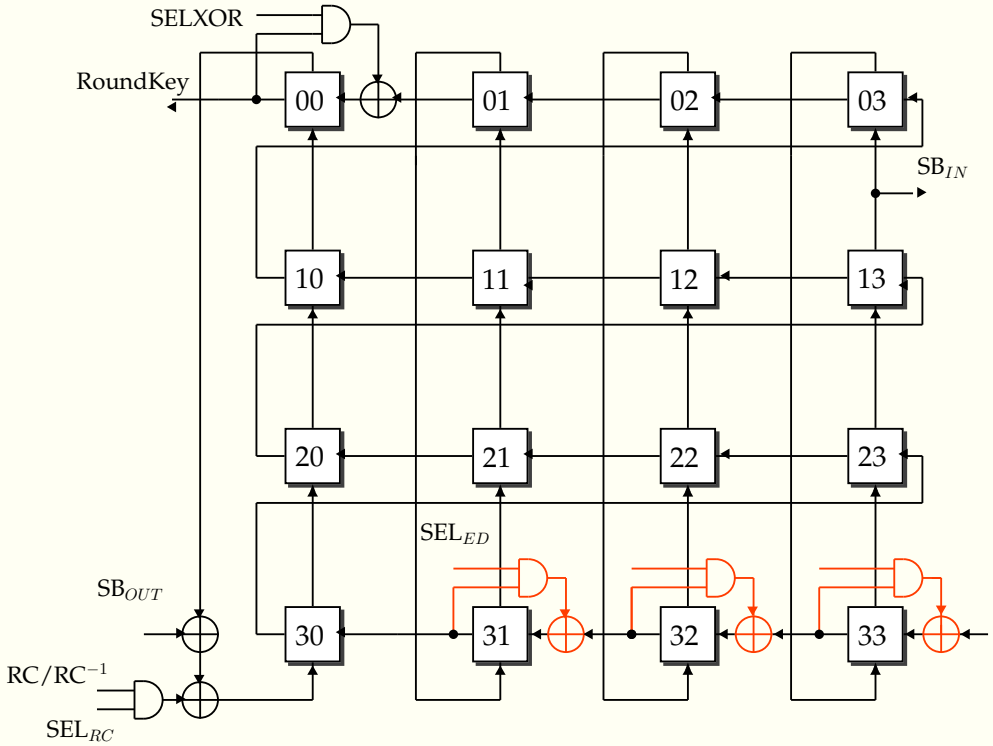
  $L_0 = K_0 \oplus F(K_3), \quad L_1 = K_1 \oplus L_0, \quad L_2 = K_2 \oplus L_1, \quad L_3 = K_3 \oplus L_2$
- For Decryption, roundkeys generated in reverse order

  Given L_0, L_1, L_2, L_3 we need to generate K_0, K_1, K_2, K_3.

$$
\begin{aligned}
K_3 &= L_2 \oplus L_3 \\
K_2 &= L_1 \oplus L_2 \\
K_1 &= L_0 \oplus L_1 \\
K_0 &= F(K_3) \oplus L_0 = F(L_2 \oplus L_3) \oplus L_0
\end{aligned}
$$

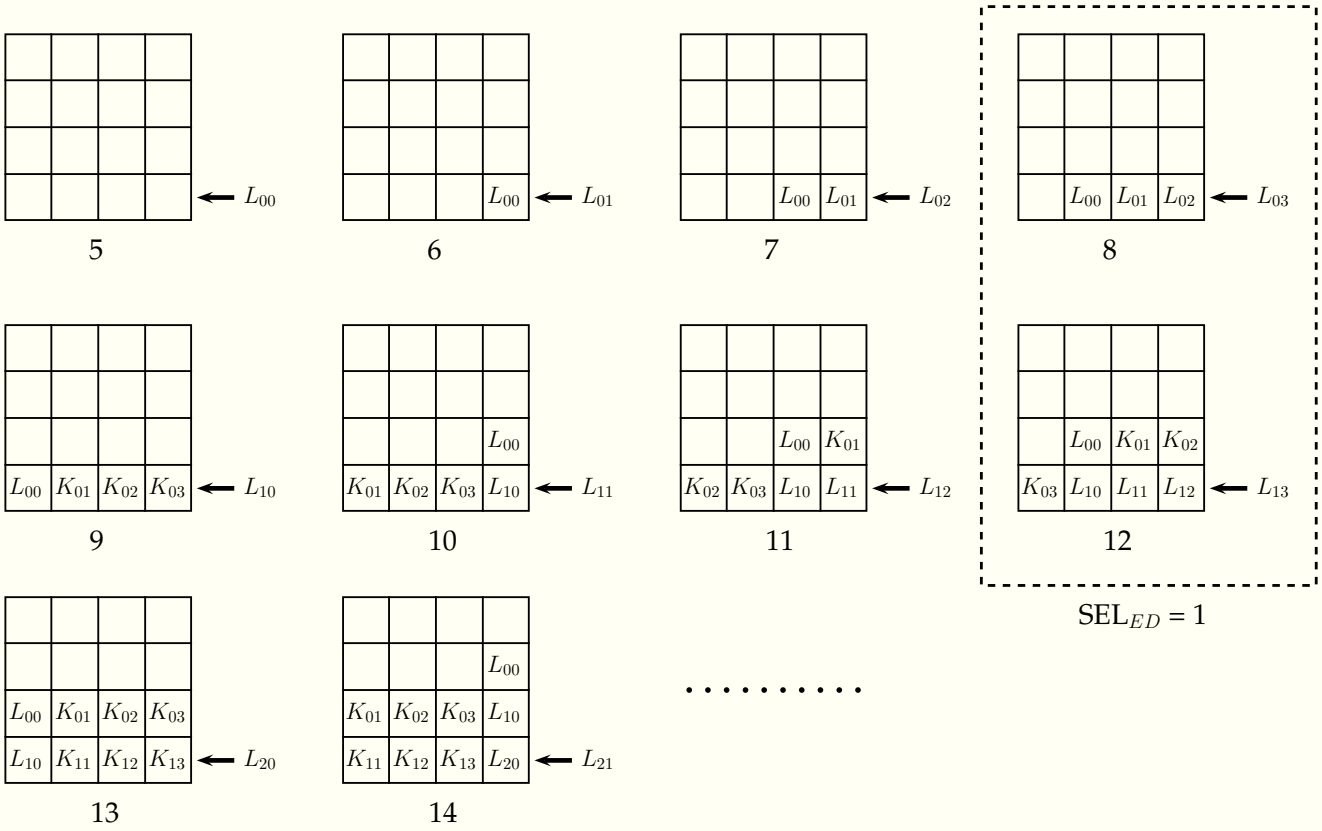
- What modification to Key register circuit is needed ???

Principal Issues: Inv. Keyschedule



- How does this help ???

Principal Issues: Inv. Keyschedule

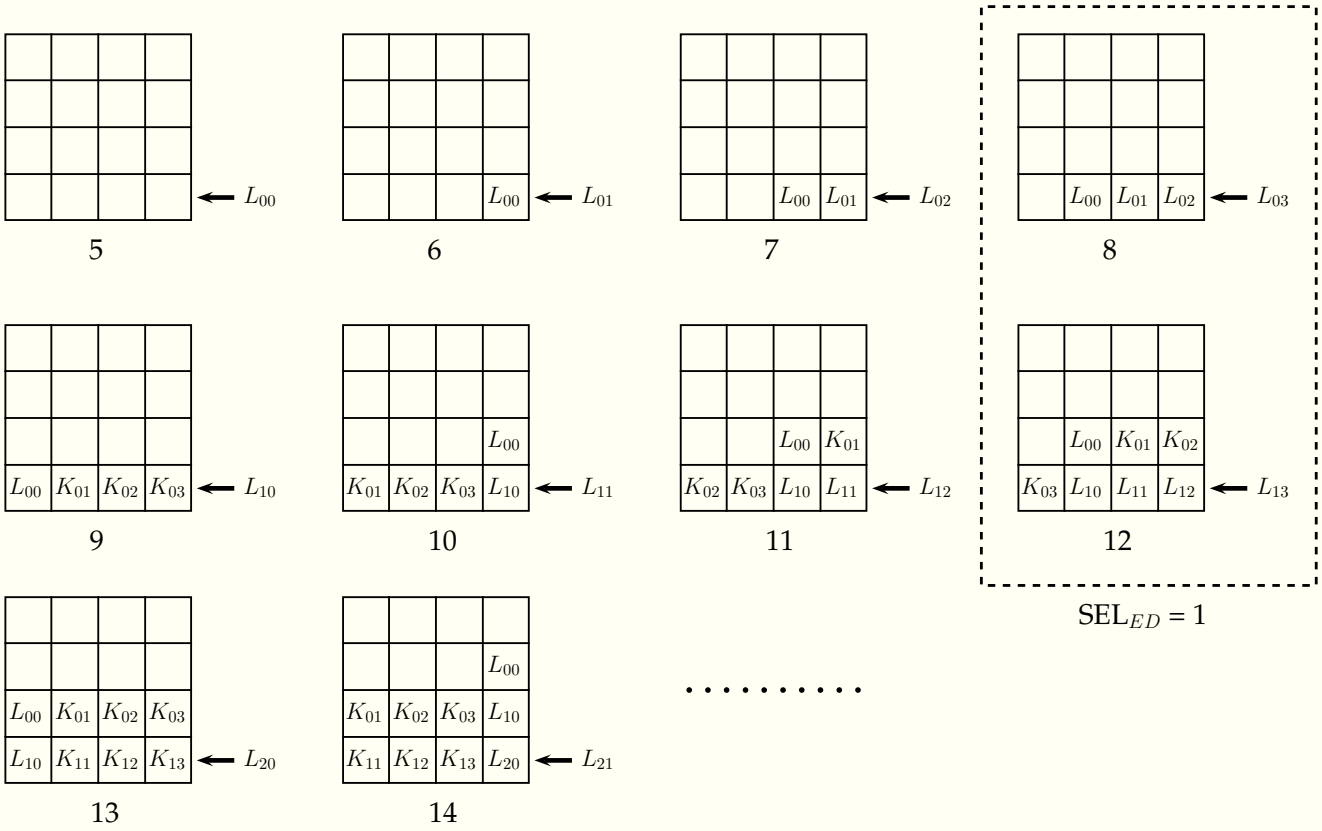


- Let $L_{0i}, L_{1i}, L_{2i}, L_{3i}$ denote the 4 key bytes in the column L_i

  Set SEL$_{ED}$ to 1 only during cycles 8, 12, 16, 20.

  Serially load the key bytes from 5-20.

Principal Issues: Inv. Keyschedule



- After cycle 20, the Key register contains L_0, K_1, K_2, K_3.

  Compute $F(K_3)$ in cycles 1-4 as during encryption and add to L_0.

  At the beginning of next cycle 5, entire roundkey is available.

Principal Issues: Operation Flow

1. Add whitening key.
2. Rounds 1 to 9
   A. Sub. Layer, **B.** Shiftrows, **C.** Mixcolumn, **D.** Add roundkey
3. Round 10
   A. Sub. Layer, **B.** Shiftrows, **C.** Add roundkey

Encryption Round

Shiftrow $\rightarrow$ Mixcolumn $\rightarrow$ Add roundkey + S-box of next round

Principal Issues: Operation Flow

1. Add whitening key.
2. Rounds 1
   A. Inv Sub. Layer, **B.** Inv Shiftrows, **C.** ARK
3. Round 2-10
   A. Inv Mixcolumn, **B.** Inv Sub. Layer, **C.** Inv Shiftrows, **D.** ARK

Decryption Round ??

$$\text{Shiftrow}^{-1} \rightarrow \text{Mixcolumn}^{-1} \rightarrow \text{ARK} + \text{S-box}^{-1} \text{ of next round}$$

- Used in Satoh [AC 01]. Reverses order of $\text{MC}^{-}1$ and ARK.
  Requires $MC^{-1}(K)$ to work $\rightarrow$ Additional Time or Latency

Principal Issues: Operation Flow

1. Add whitening key.
2. Rounds 1
   A. Inv Sub. Layer, **B.** Inv Shiftrows, **C.** ARK
3. Round 2-10
   A. Inv Mixcolumn, **B.** Inv Sub. Layer, **C.** Inv Shiftrows, **D.** ARK

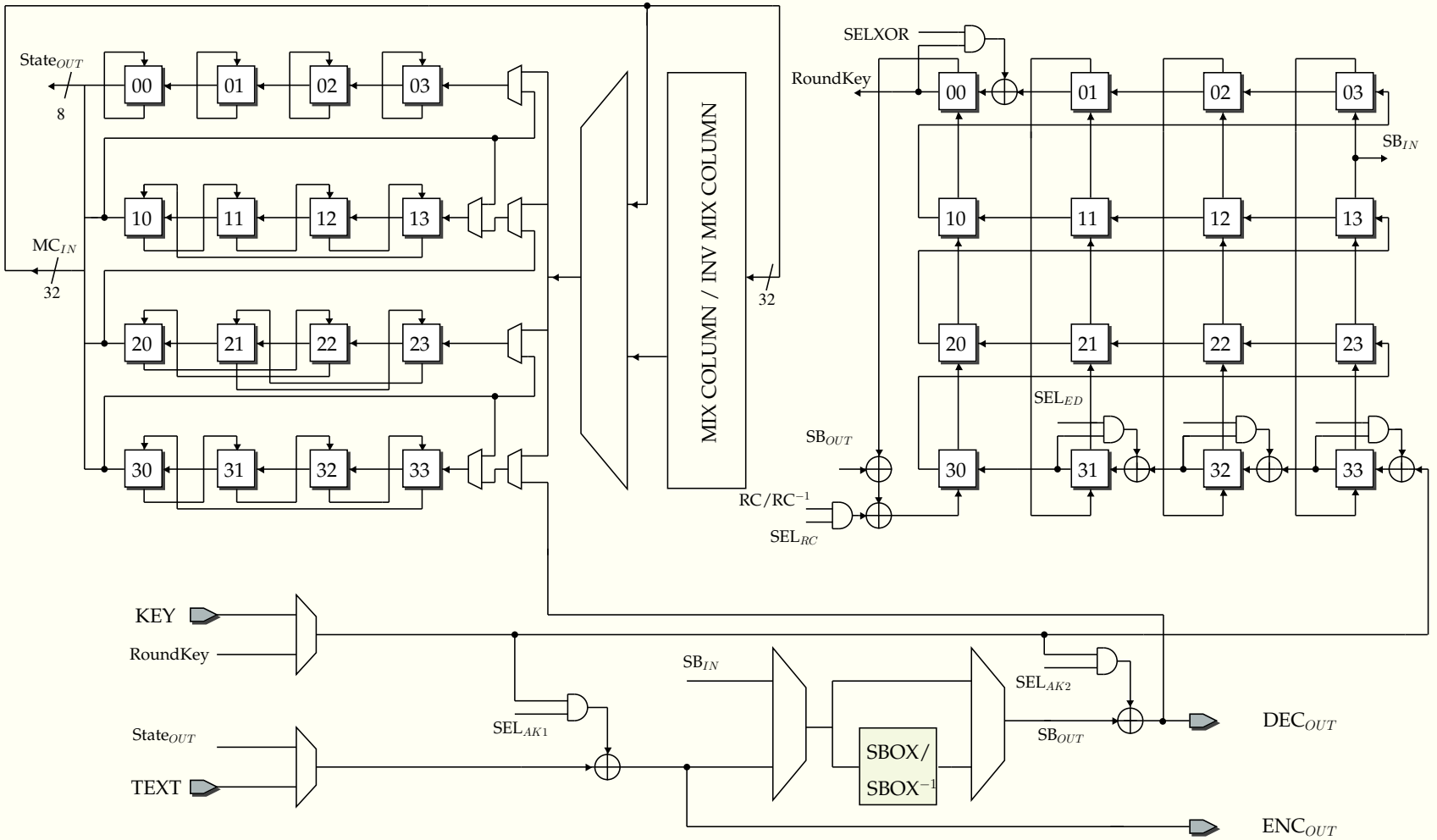
Decryption Round ??

$$\text{Mixcolumn}^{-1} \rightarrow \text{Shiftrow}^{-1} \rightarrow \text{S-box}^{-1} + \text{ARK}$$

- Mirror inverse of Encryption round.

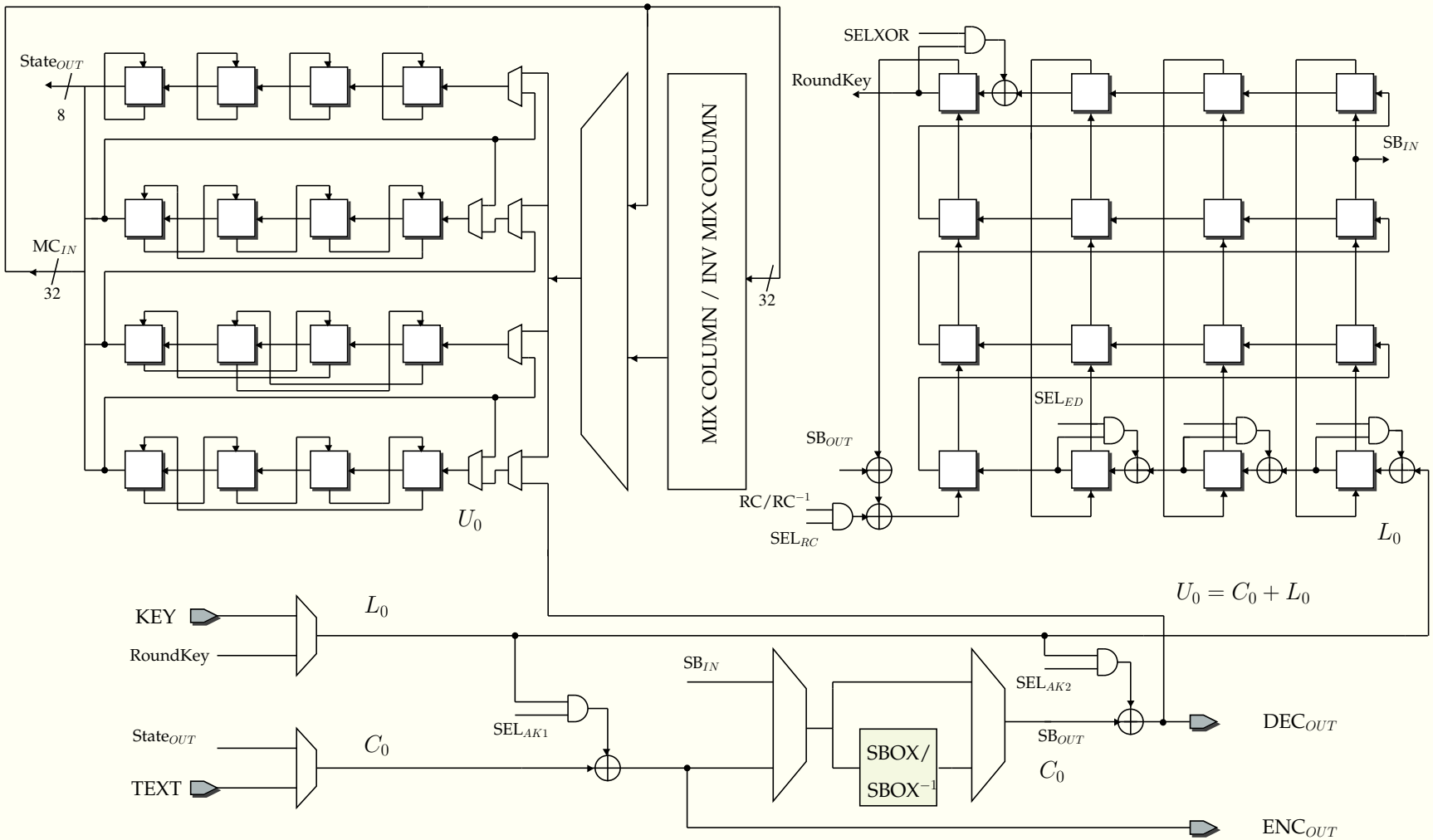
  $MC^{-1}(K)$ not required.

Atomic-AES


State$_{OUT}$
8
00
01
02
03
10
11
12
13
MC$_{IN}$
32
20
21
22
23
30
31
32
33
MIX COLUMN / INV MIX COLUMN
32
SELXOR
RoundKey
SB$_{IN}$
SEL$_{ED}$
SB$_{OUT}$
RC/RC^{-1}
SEL$_{RC}$
KEY
RoundKey
State$_{OUT}$
TEXT
SEL$_{AK1}$
SB$_{IN}$
SBOX/
SBOX^{-1}
SEL$_{AK2}$
SB$_{OUT}$
DEC$_{OUT}$
ENC$_{OUT}$

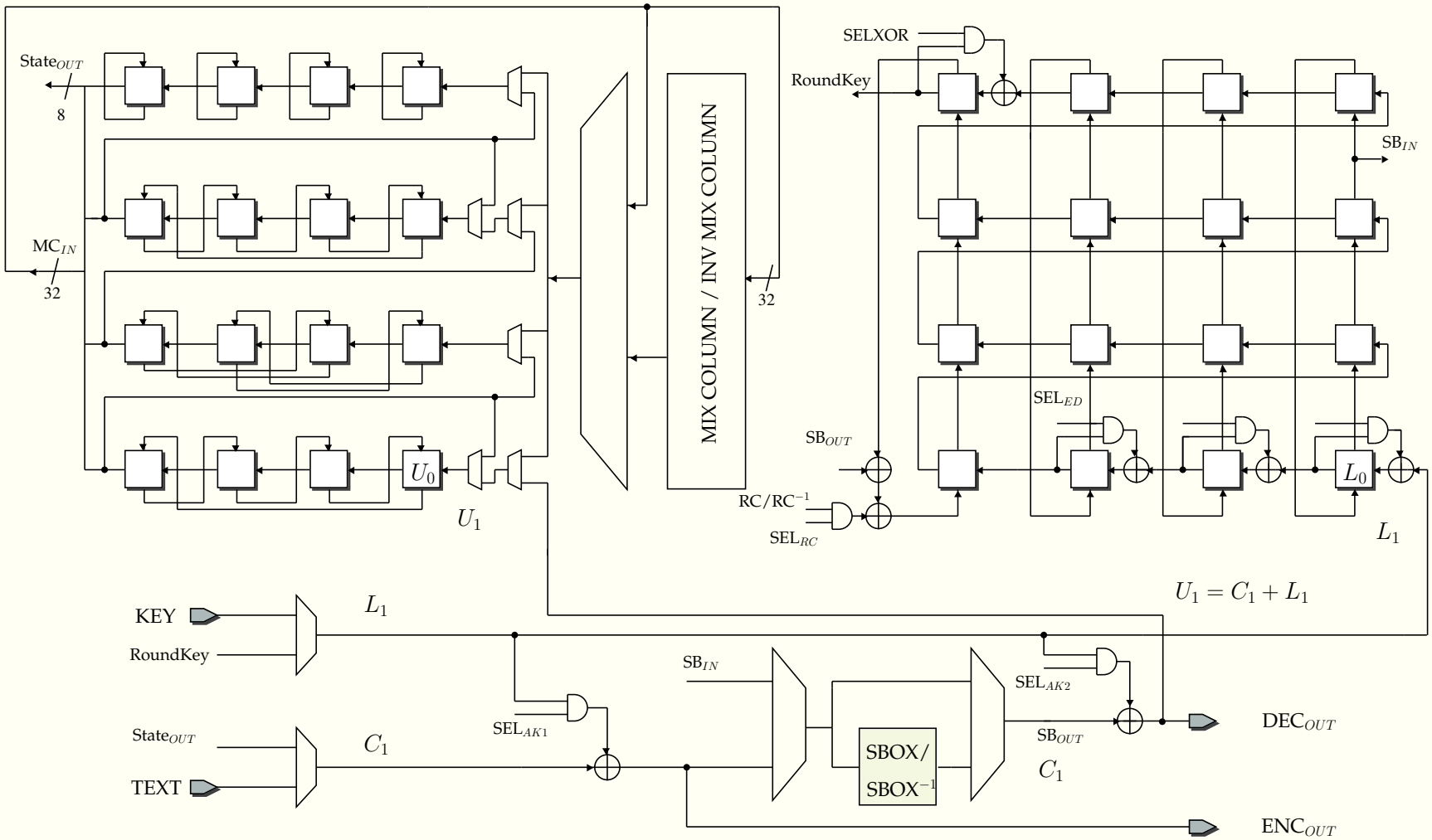
Data flow

Round 0, Cycle 5



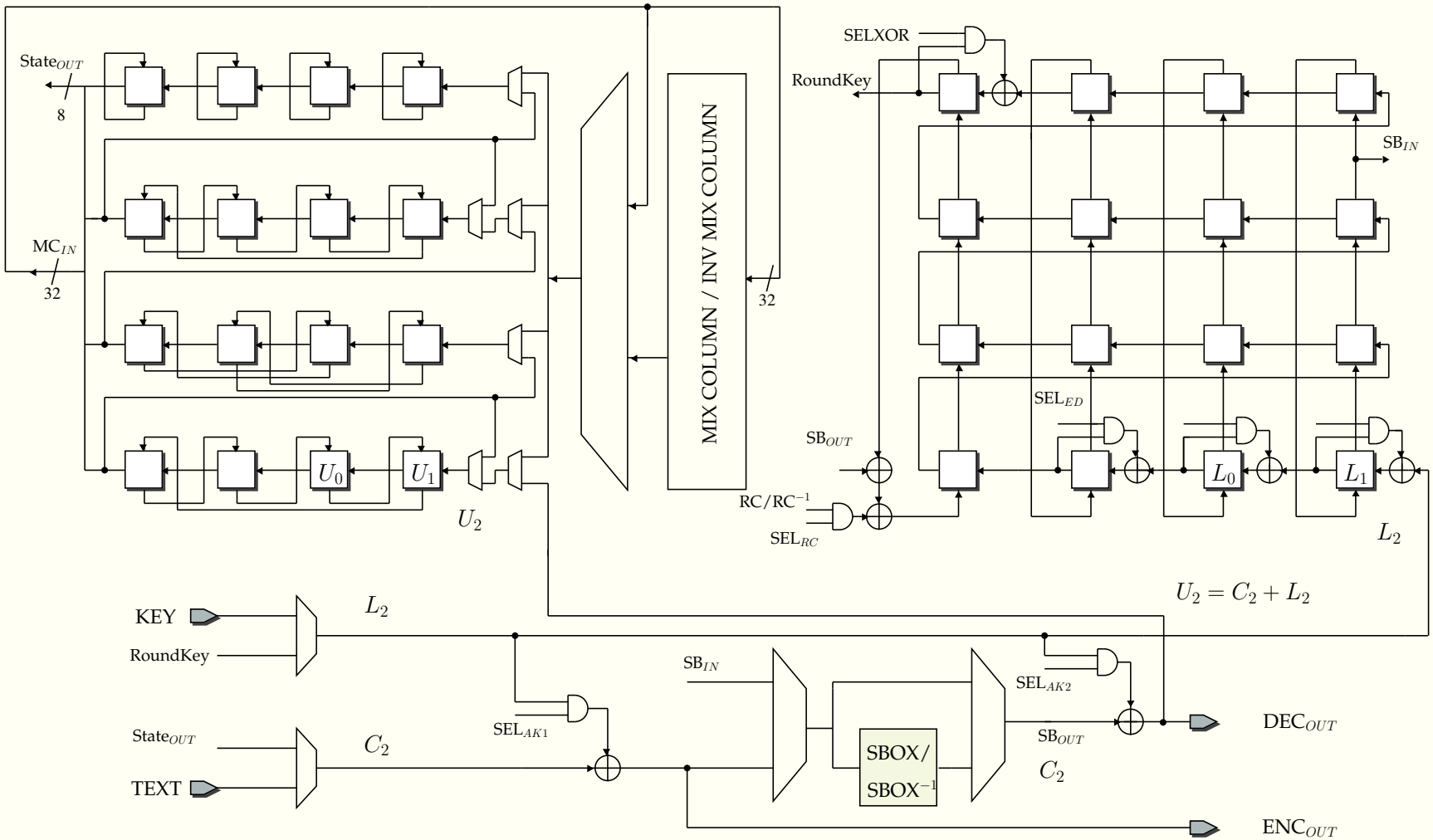
Data flow

Round 0, Cycle 6



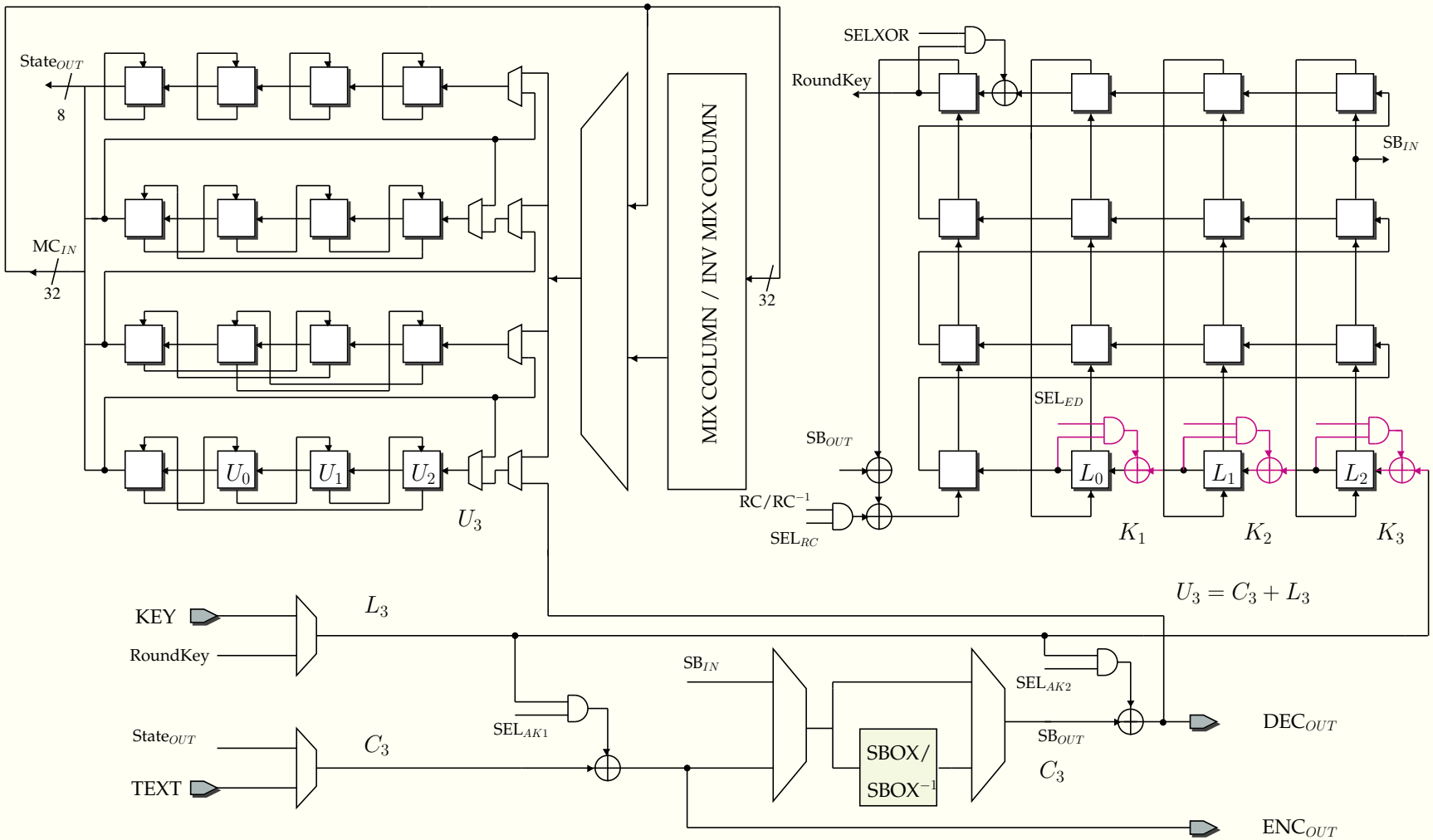
Data flow

Round 0, Cycle 7



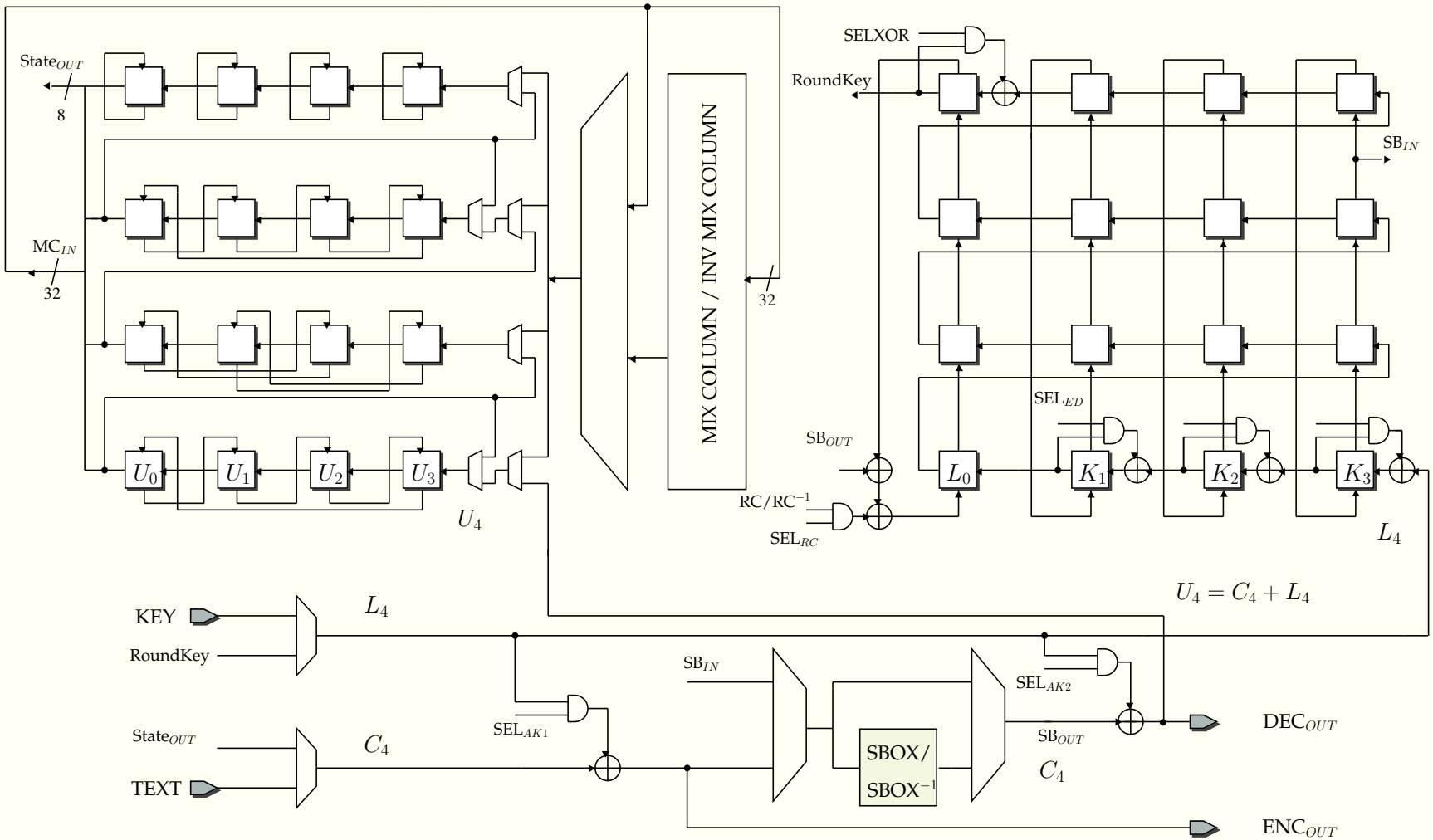
Data flow

Round 0, Cycle 8



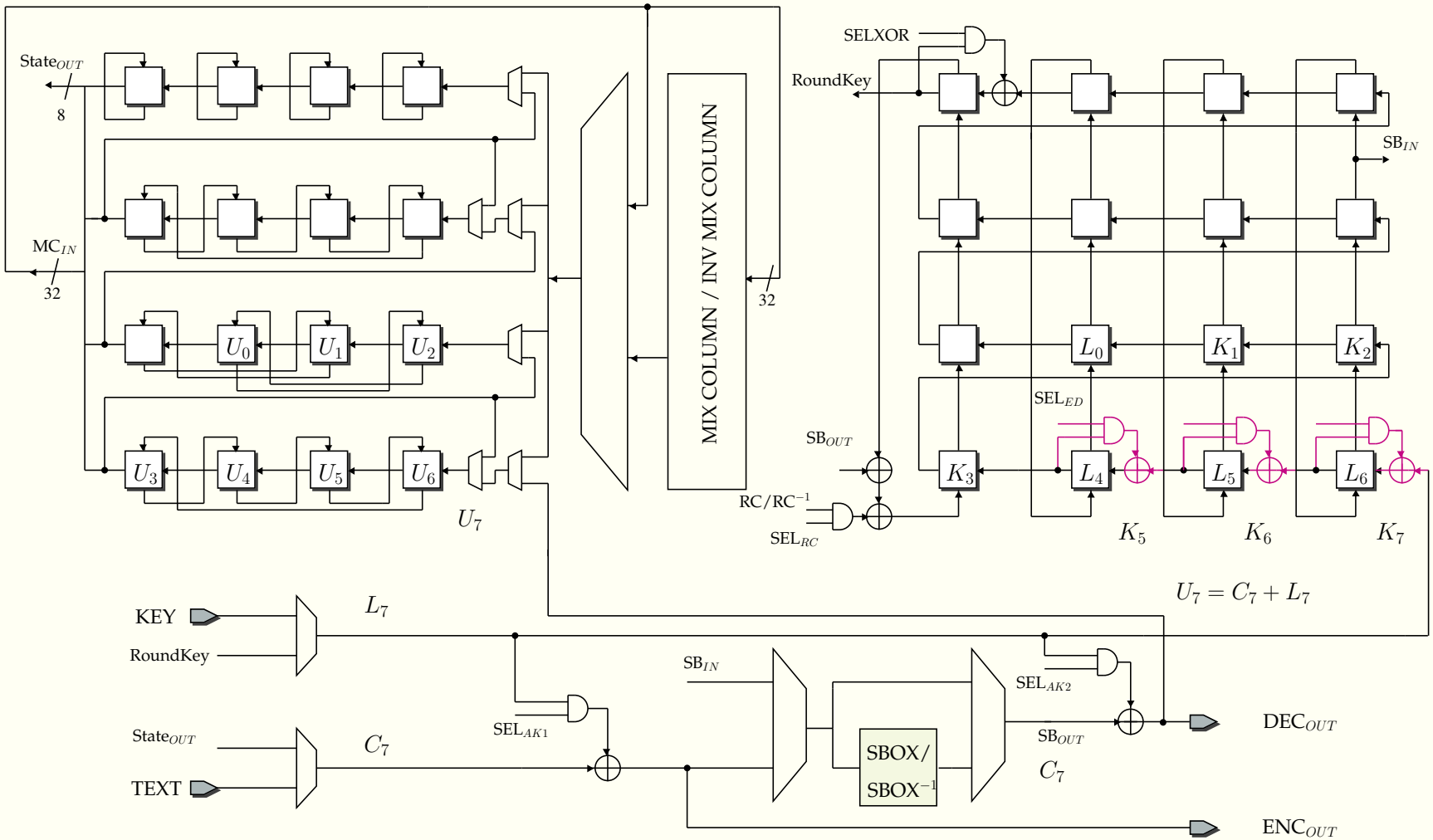
Data flow

Round 0, Cycle 9



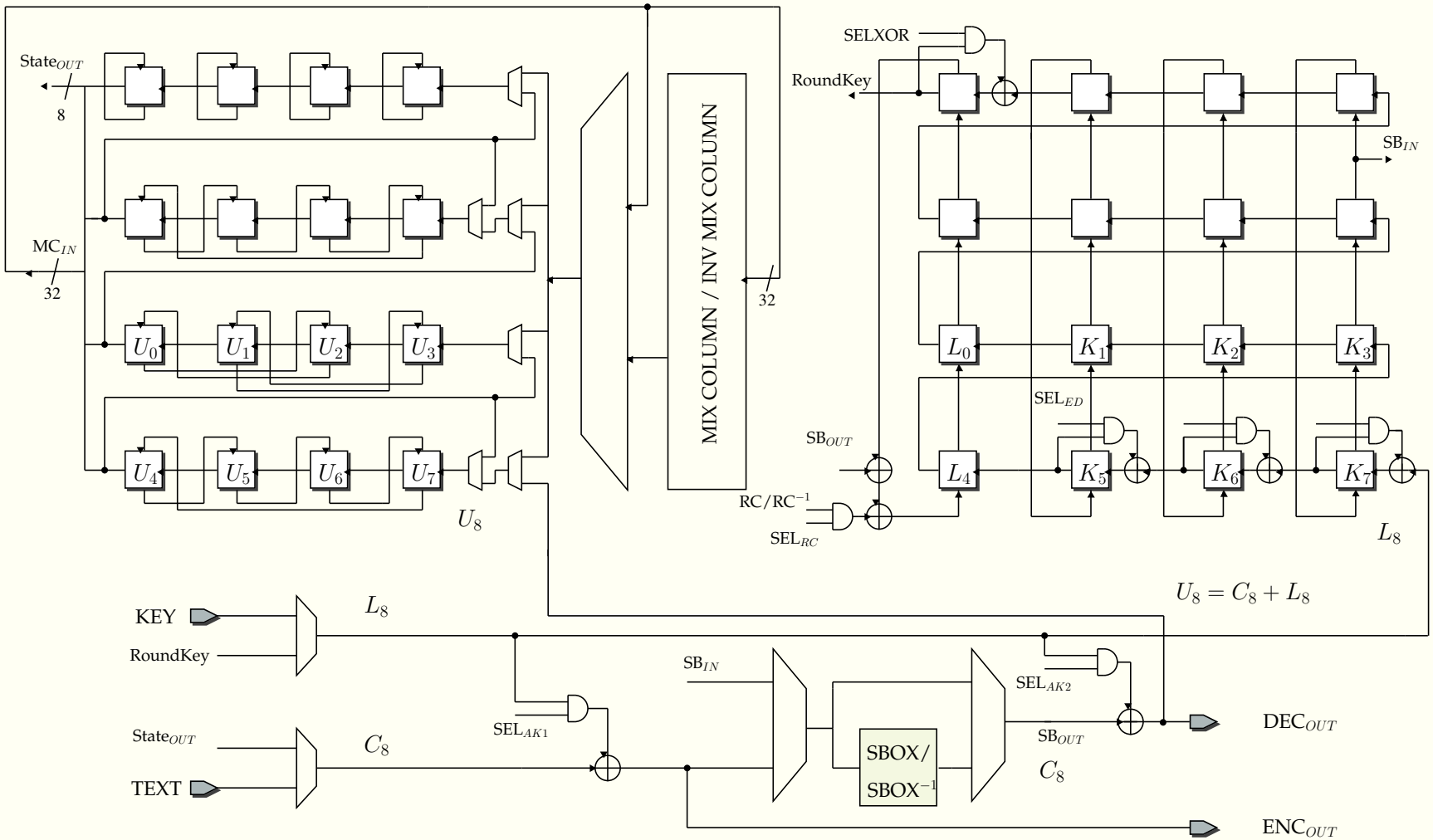
Data flow -IMC-ISR-ISB-ARK

Round 0, Cycle 12



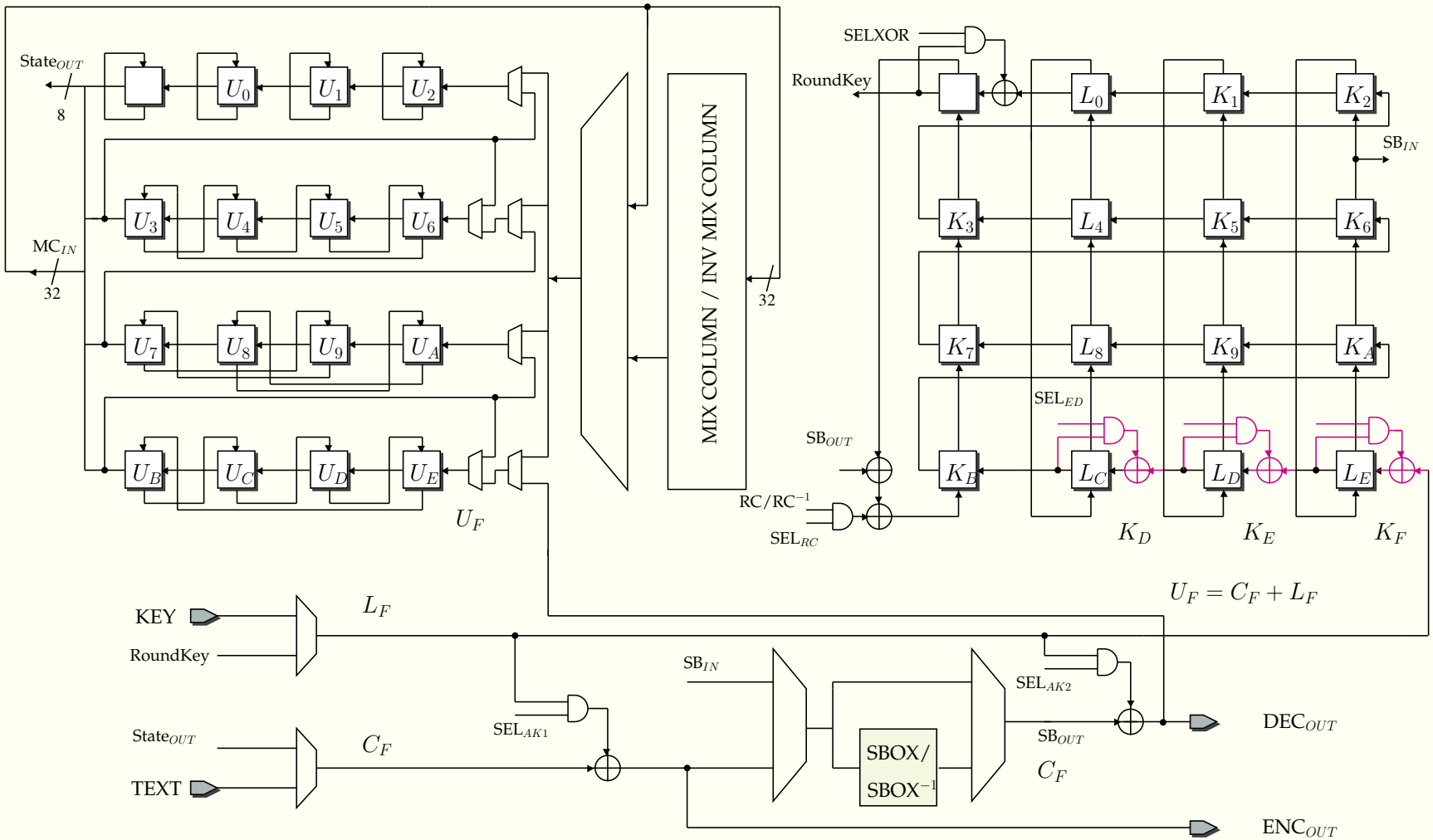
Data flow -IMC-ISR-ISB-ARK

Round 0, Cycle 13



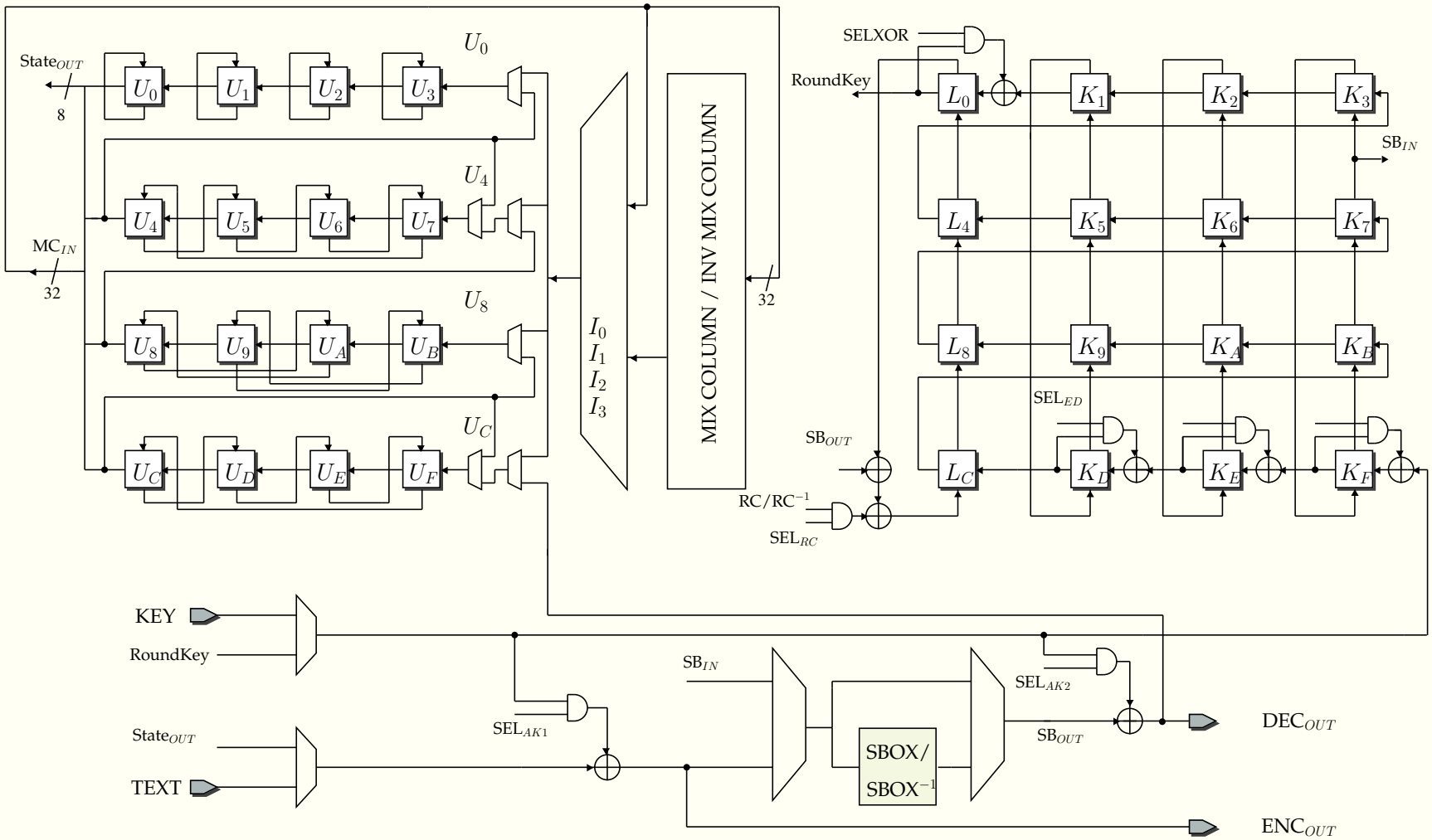
Data flow -IMC-ISR-ISB-ARK

Round 0, Cycle 20



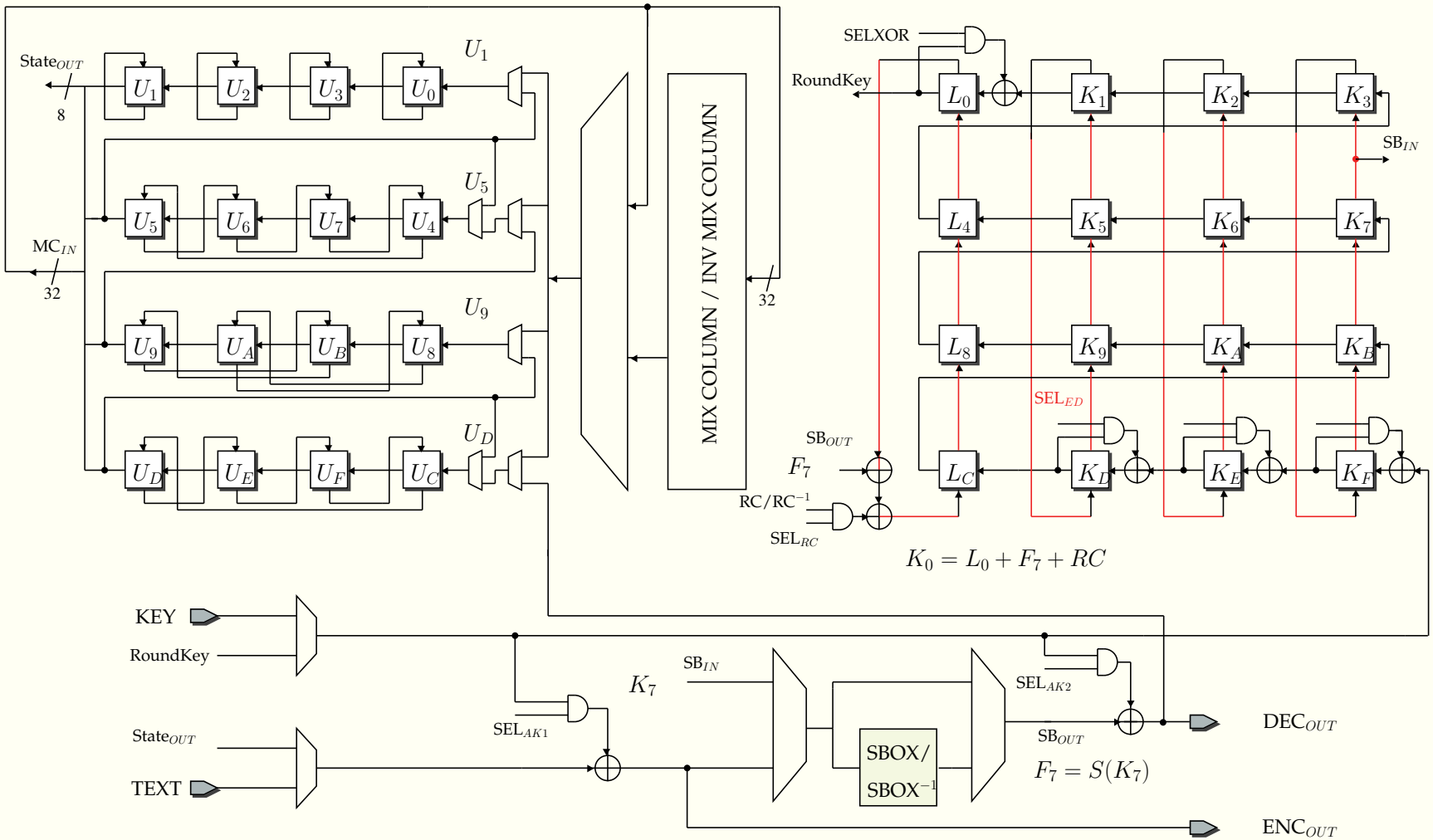
Data flow -IMC-ISR-ISB-ARK

Round 1, Cycle 0



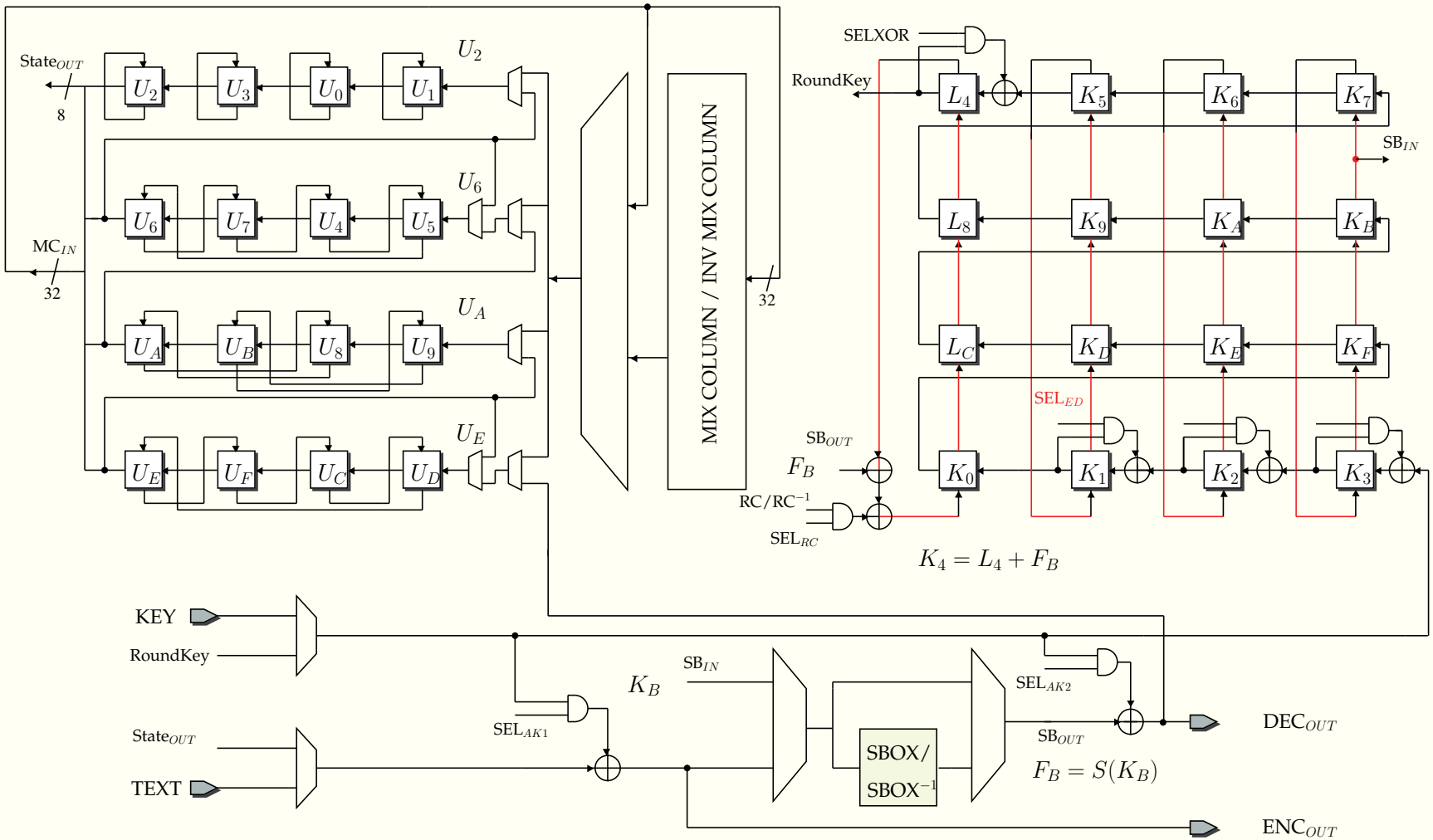
Data flow -IMC-ISR-ISB-ARK

Round 1, Cycle 1



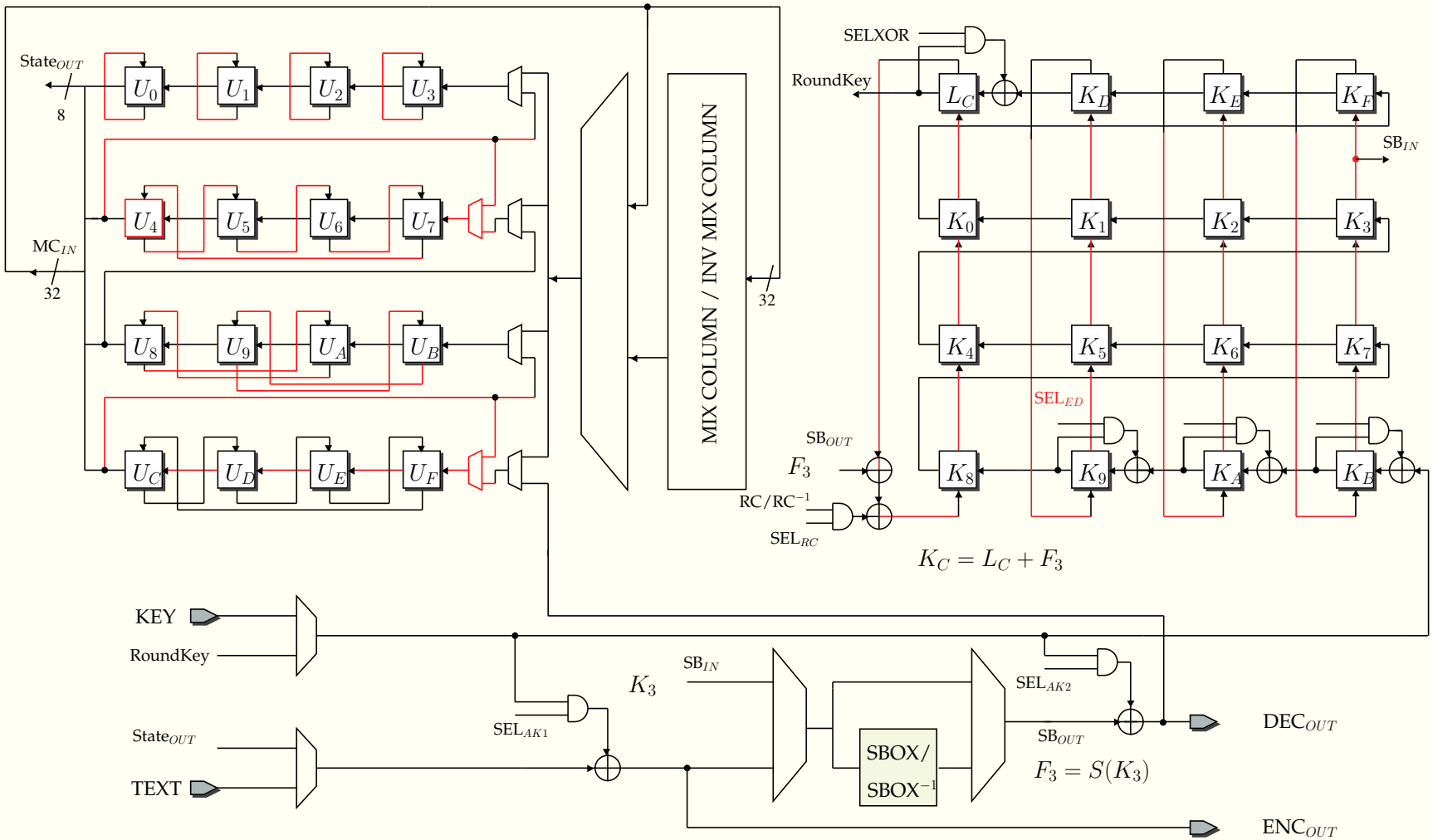
Data flow -IMC-ISR-ISB-ARK

Round 1, Cycle 2



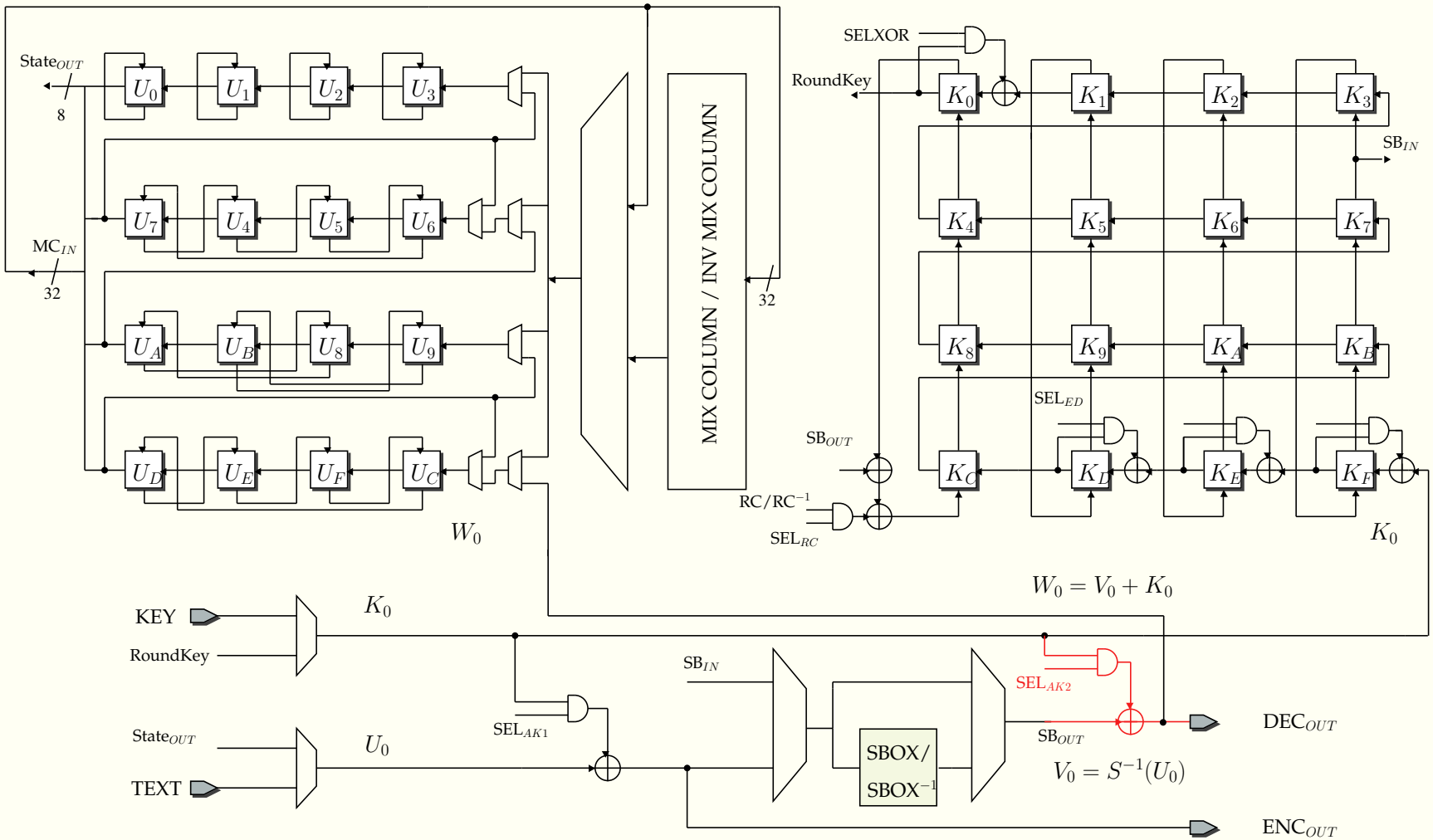
Data flow -IMC-ISR-ISB-ARK

Round 1, Cycle 4



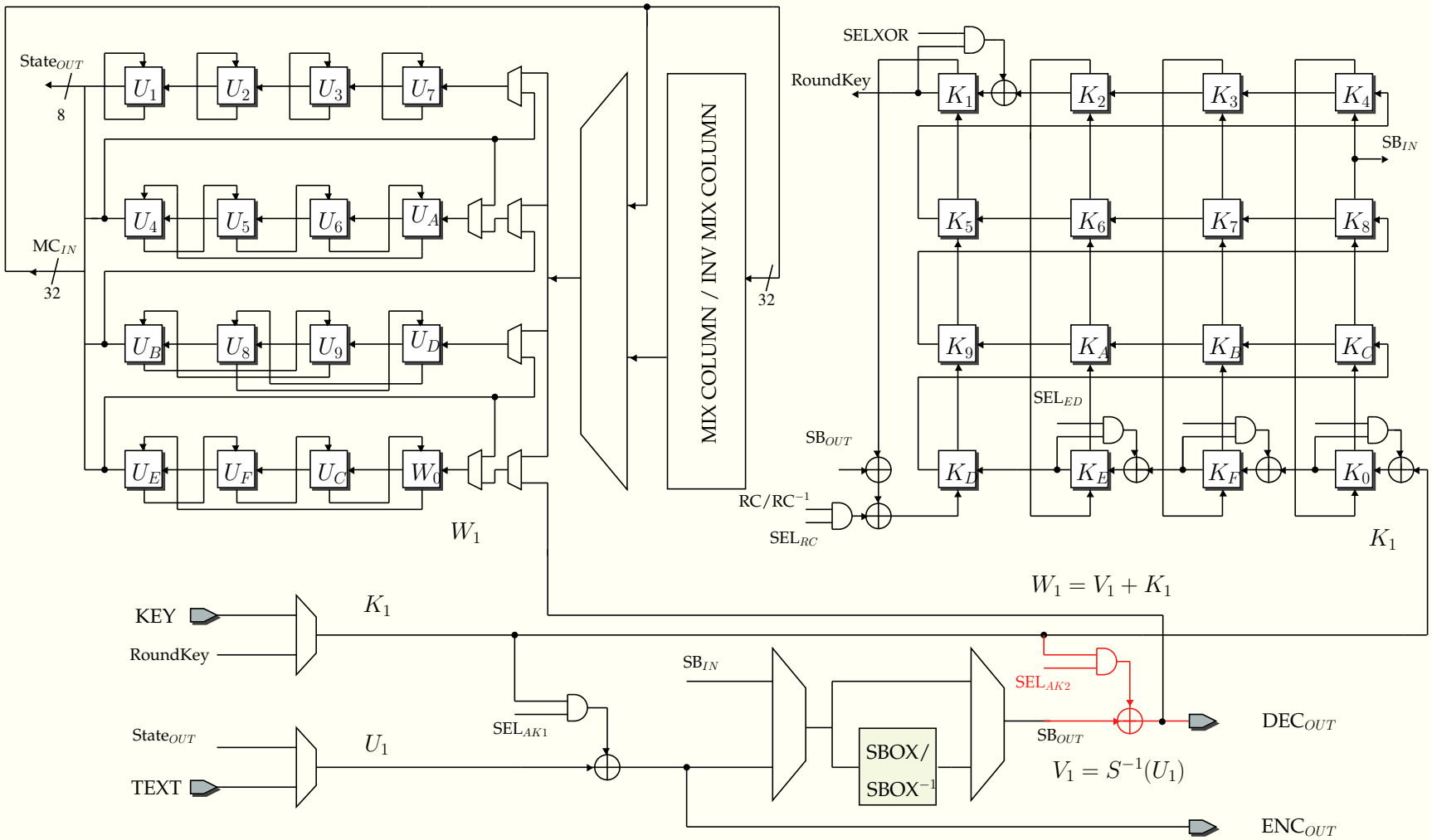
Data flow -IMC-ISR-ISB-ARK

Round 1, Cycle 5



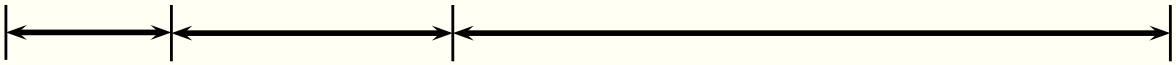
Data flow -IMC-ISR-ISB-ARK

Round 1, Cycle 6



Atomic-AES: Operations

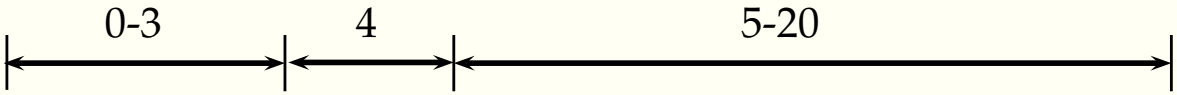
ENCRYPTION



Round				
0	**State**			Add Whitening Key + S-box of 1st round
	Key			Store Key serially
1-10	**State**	Shiftrow	Mixcolumn	Add roundkey + S-box of next round
	Key	Frozen	Compute $F(K_3)$	Compute roundkey + Store it serially

DECRYPTION

Round				
0	**Key**			Store Key serially (with SEL_{ED}=1 at 8,12,16,20)
1-10		Frozen	Compute $F(K_3)$	Store Key serially (with SEL_{ED}=1 at 8,12,16,20)
0	**State**			Add Whitening Key
1-10		Mixcolumn^{-1}	Shiftrow^{-1}	Inverse S-box + Add roundkey



Atomic-AES: Additions

1. 2 additional 8-bit multiplexers in the state datapath,
2. 3 additional 8-bit xor gates in the key datapath,
3. 24 additional and gates in the key datapath,
4. 1 additional 8-bit multiplexer, 1 additional 8-bit xor gate, 16 additional and gates during state-key addition,
5. Other additional logic required to implement
    a. S-box and its inverse,
    b. Mixcolumn and its inverse,
    c. Round constants and their inverses.

S-box

- A discussion on AES S-box architectures can take over 5 hours.
- Most use tower field representations of $GF(2^8)$
- Architecture proposed by Canright [CHES 04]

  Using normal bases to represent $GF(2^8)$

  One of the most compact representations of Rijndael S-box.
- Improved by Boyar-Peralta [JOC 11] (Forward S-box)
- We use this architecture.

Mixcolumn + Inverse

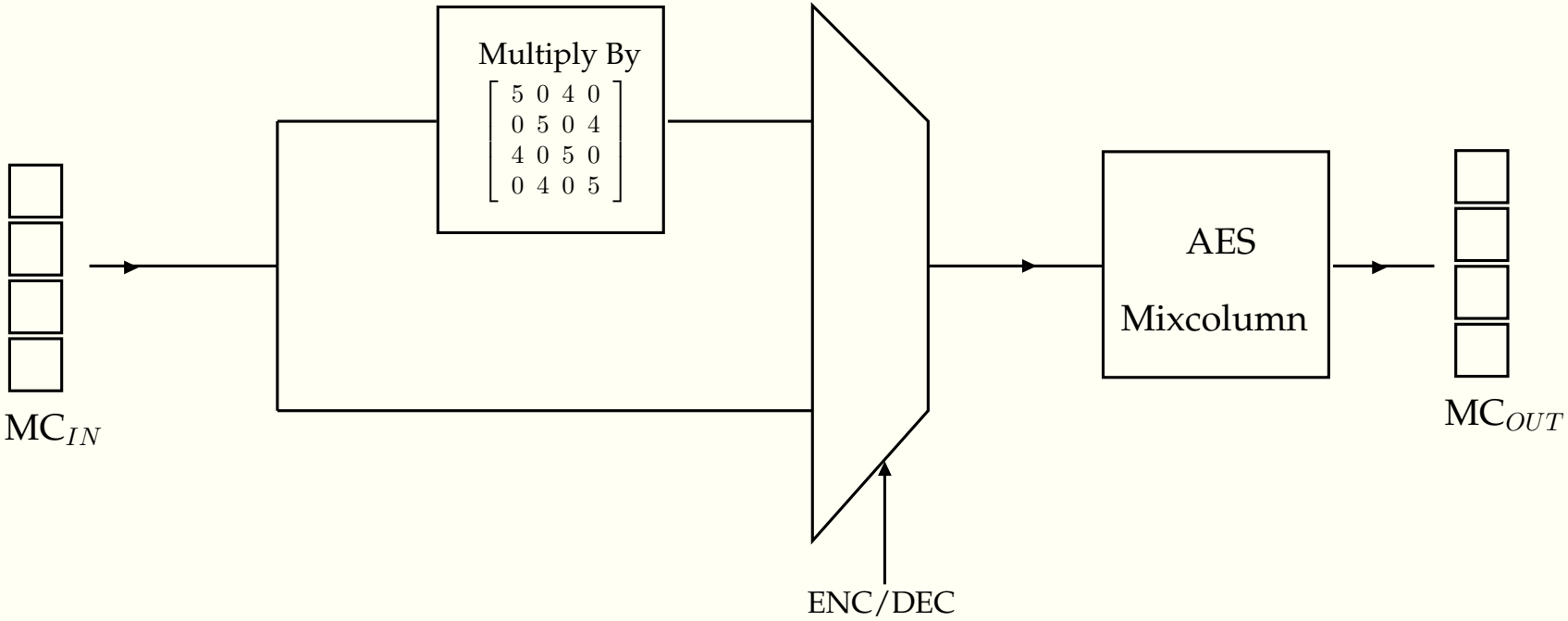
$$\begin{pmatrix} 14 & 11 & 13 & 9 \\ 9 & 14 & 11 & 13 \\ 13 & 9 & 14 & 11 \\ 11 & 13 & 9 & 14 \end{pmatrix} = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} 5 & 0 & 4 & 0 \\ 0 & 5 & 0 & 4 \\ 4 & 0 & 5 & 0 \\ 0 & 4 & 0 & 5 \end{pmatrix}$$

- Premultiply of input column by the Circulant$(5, 0, 4, 0)$:

$$y_3 = \texttt{xxtime}(x_3 \oplus x_1) \oplus x_3, \quad y_2 = \texttt{xxtime}(x_2 \oplus x_0) \oplus x_2$$

$$y_1 = \texttt{xxtime}(x_3 \oplus x_1) \oplus x_1, \quad y_0 = \texttt{xxtime}(x_2 \oplus x_0) \oplus x_0$$

Mixcolumn + Inverse



- The multiplication block takes exactly 58 xor gates
- Mixcolumn takes 108 gates.
- Entire ckt in $108 + 58 = 166$ gates and 32 bit multiplexer.

Other features

- Round constants implemented using LUTs.
- If r is current round then:

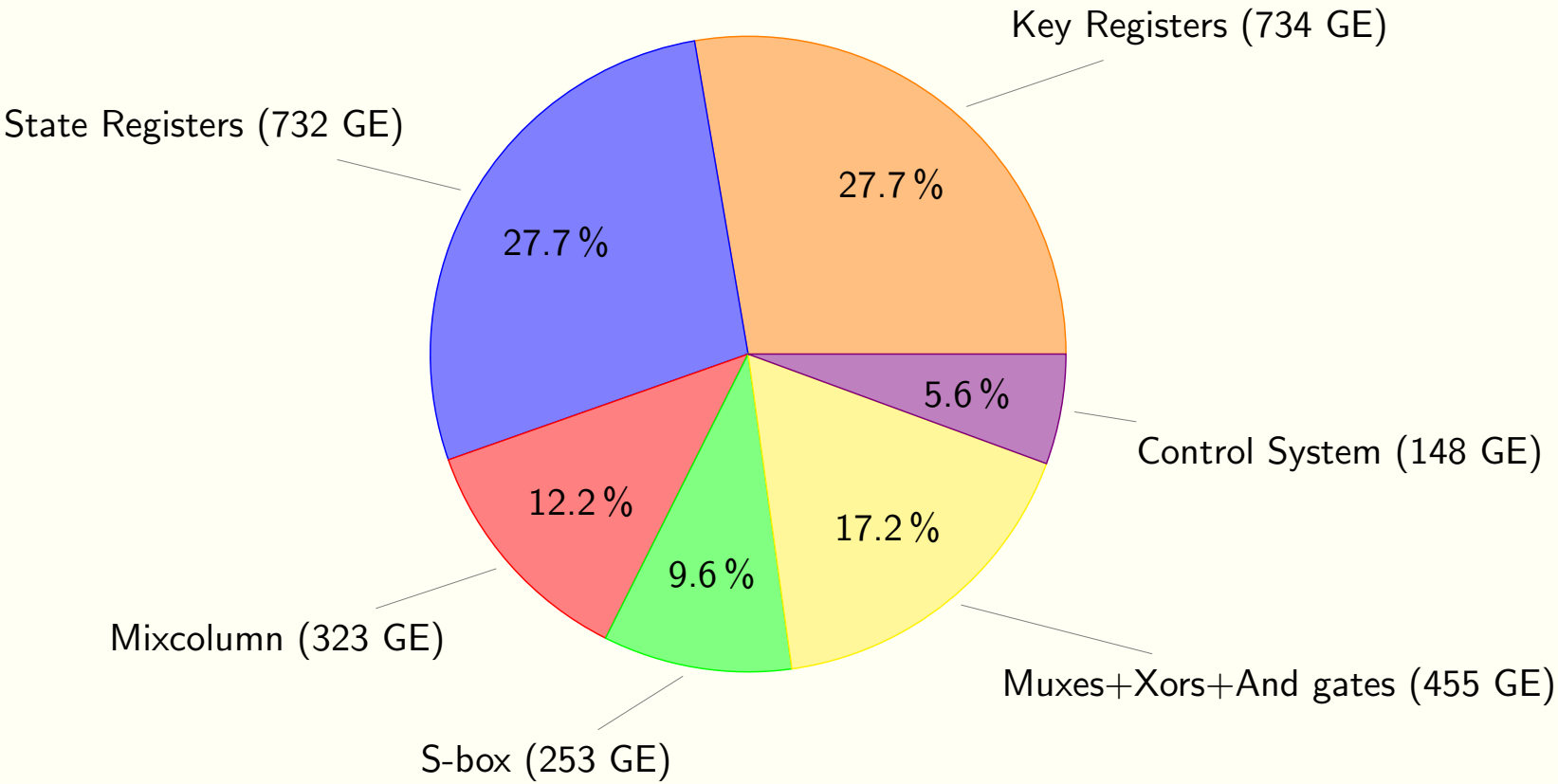
  ENC_{RC}: LUT(r), DEC_{RC}: LUT($11 - r$)
- Use a 4-bit mux to filter r or $11 - r$ and input to LUT.
- All signals generated using 21 cycle LFSR.

Implementation results

#	Architecture	Type	Library	Area (GE)	Latency		Energy (in nJ)	Max Throughput (Mbps)
					ENC	DEC		
1	8-bit Serial (EC11)	ENC only	UMC 180nm	2400	226	-	8.4	-
2	Grain of Sand (IEE-IS05)	ENC/DEC	Philips 350nm	3400	1032	1165	46.4/52.4	9.9/8.8
3	8-bit Serial (IEEE JSSC15)	ENC/DEC	22nm	4037	336	216	3.9/2.5	432.0/671.0
4	32-bit Serial (AC01)	ENC/DEC	110nm	5400	54	54	-	311.0
5	Atomic-AES	ENC/DEC	STM 90nm	**2645**	226	226	3.3	94.4
		ENC/DEC	STM 65nm	2976	226	226	2.2	57.8

Performance Comparison

Implementation results



Area requirements of the individual components

Thank you for listening!! Any Questions??