Politecnico di Milano
Academic year 2020-2021

EMBEDDED SYSTEMS

Embedded systems project

AUBRY Julie
LUKACEVIC Rebeka
Milano

March 23, 2021

Contents

1 Introduction

The aim of the project is the following: Implementation of the Cipher Block Chaining (CBC) Advanced Encryption Standard (AES) algorithm on the platform containing a Cortex-M3 microprocessor, an AXI interface, a BRAM memory and GPIOs for LEDs and push buttons using Vivado for hardware and µVision for software design.

Development of the project started from the Cortex-M3 Design Start FPGA-Xilinx edition provided by Arm. In order to create customized project, modifications were applied on both hardware and software parts. In the following sections detailed information of aforementioned actions will be described.

2 Hardware

Goals of hardware adjustments were to clean out provided scheme, retain and/or append elements/blocks and ports necessary for proper functioning of the application.

2.1 Changes

Starting point was deleting unused blocks. Firstly, *AXI_quad_spi* (Quad Serial Peripheral Interface) and DAPlink because the optional V2C-DAPLink adaptor board that provides a debug flow is not used. Secondly, *axi_gpio_1* because only one GPIO block is sufficient and *axi_uartlite_0* as it is not used. Thirdly and finally, *tri_io_buf_o* block and consequently dangling ports were removed as they were marked as needless in this case. Reasoning behind is that IOBUF is bi-directional buffer with the concept of third, high impedance state used to effectively remove device's influence from the rest of the circuit.

Furthermore, changes on the remaining blocks were done to match previous modifications.

Clocks and Resets block: Since DAPlink was removed because it was marked as unnecessary, accompanying actions were done on this block deleting linked blocks and ports or adding new connections and external ports.

AXI_Interconnect: The number of output ports has been reduced to 2 in order to communicate with the AXI_GPIO and the AXI_BRAM_Controller.

CFGITCMEN[1:0]: Through reduced blocks *Concat* and *Constant* this port is set to fullfill desired requirements.

- CFGITCMEN[0] is set indicating that there is no V2C-DAPLink board and the internal RAM ITCM is mapped to the lower address alias in the memory map.
- CFGITCMEN[1] is not set meaning that the internal RAM ITCM is not mapped to the upper address alias in the memory map.

IRQ[7:0]: compressed to [1:0] and through aligned *Concat* and *Constant* block set in order to enable interrupts of present GPIO.

2.2 Memory

The Block Random Access Memory (BRAM) is connected to BRAM controller as in standard configuration which is then communicating with the processor through an AXI connection. All the data used for the CBC AES encryption algorithm are contained in a BRAM e.g. the plain text, the key, the *Sbox* and reverse *Sbox* function, the initialization vector and the cipher text once the plain text has been encrypted.

2.3 GPIOs

The board contains 4 push buttons and 4 LEDs indicated as *led_4bits* and *push_buttons_4bits* which are connected to IP interfaces GPIO and GPIO2. The communication of events from/to the GPIO to/from the microprocessor is done through an AXI interface.

Concerning the push buttons, only the first one is used. The start of an encryption/decryption of the text is triggered on the rising edge of the signal coming from the button.

Regarding the LEDs, only the 3 first are used to communicate information to the user.

- First LED:
  0: Software initialization not completed.
  1: Software initialization completed.
- Second LED:
  0: Valid BRAM content.
  1: Processing data. Invalid BRAM content.
- Third LED:
  0: Decrypted text contained in the BRAM.
  1: Encrypted text contained in the BRAM.

2.4 Simulation

In order to validate the design the following simulation has been implemented: after a delay of 8 ms enabling the software initialization, a periodic pressing of the first button occurs with a period of 6 ms including a press time of 3 ms and a release time of 3 ms. The result of the simulation are presented on the figure 1. It can be remarked that after the initialization of the software the signal *led_4bits_tri_io[0]* is risen. Also, it is perceived that on the rising edge of the *button_4bits_tri_io[0]* signal the encryption/decryption starts which is denoted with setting the *led_4bits_tri_io[1]* signal at 1. Once the operation is fullfilled and the text is encoded/decoded the *led_4bits_tri_io[1]* is reset and *led_4bits_tri_io[2]* is set/reset indicating the status of the text contained in the BRAM.

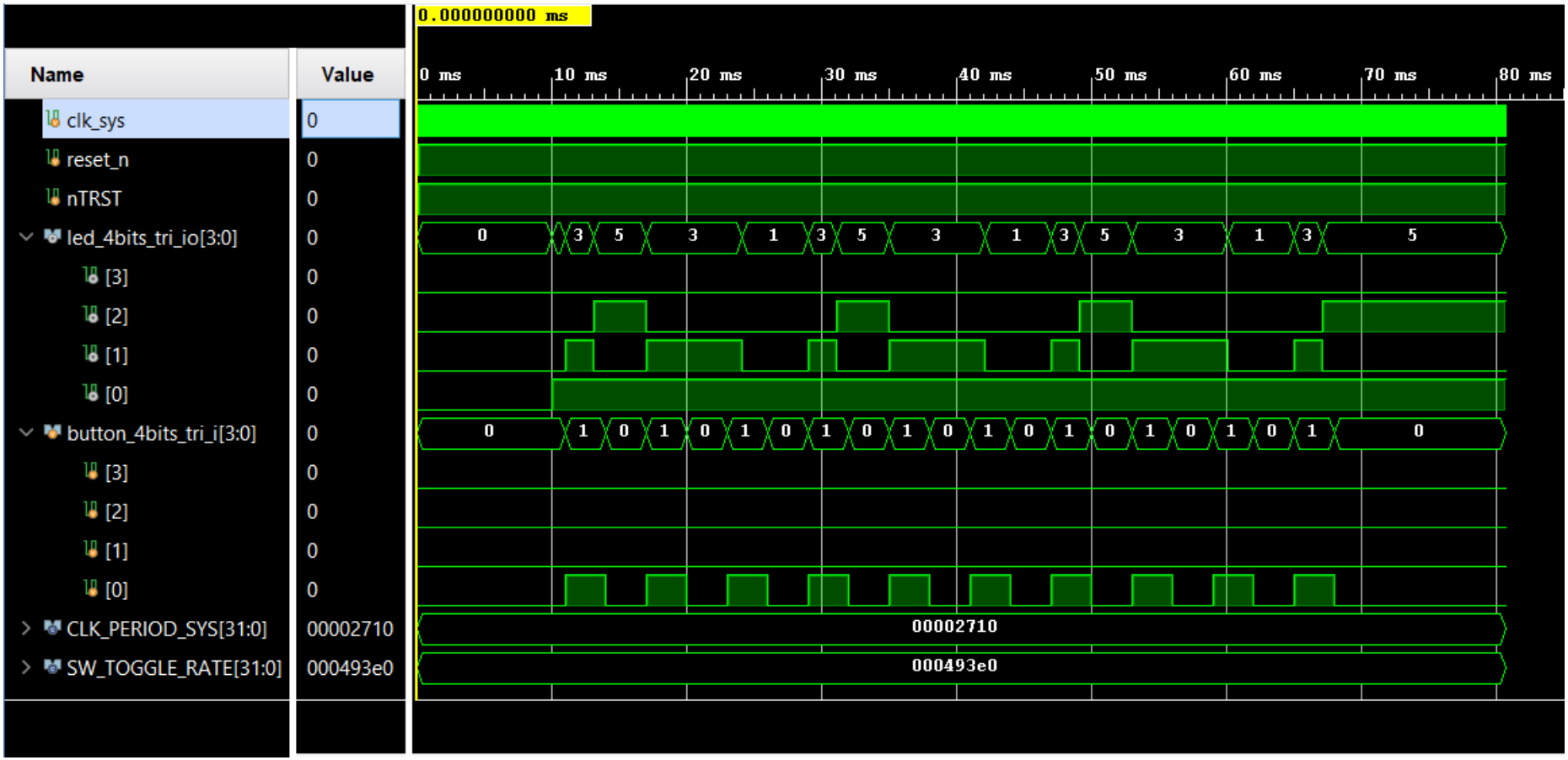

Figure 1: Simulation

3 Software

Provided project was modified in order to comply with hardware design changes and application requirements. Moreover, the core C code implementing the encryption algorithm was downloaded from link, adjusted for CBC operation mode removing non relevant lines and subsequently adapted to existing environment enabling writing and reading from memory.

Procedures of encryption and decryption are done in steps. Number of steps depends on the size of block of data and size of the key. In the table 1 N_r is the number of steps depending on number of columns of block of text N_b (as standard accepted 128 bits, which means Nb =4 bytes) and number of columns of block of the key N_k reported in bytes. The implemented algorithm is using 128 bits blocks, 128 bits key and 10 rounds.

N_r	N_b=4 bytes
N_k=4 bytes	10
N_k=6 bytes	12
N_k=8 bytes	14

Table 1: Number of steps N_r depending on N_b and N_k

For operating in the CBC mode the algorithm requires a key and an initialization vector which are created randomly.

3.1 Encryption

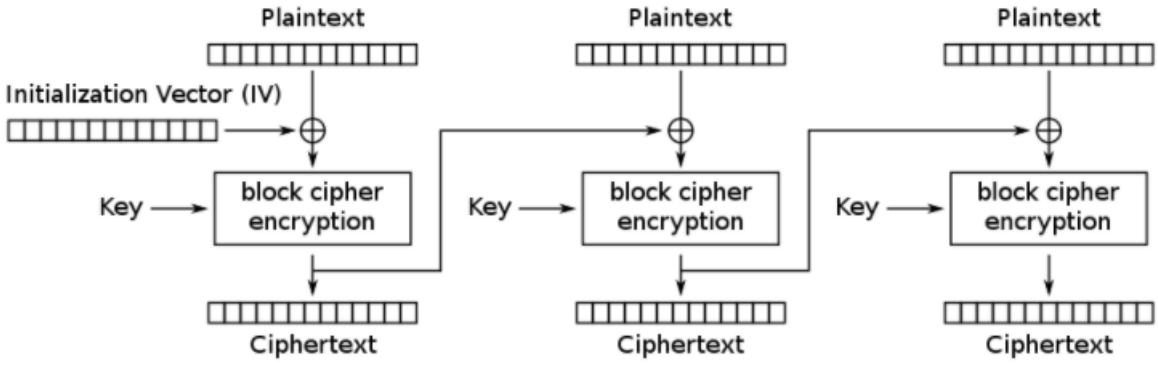

Figure 2: AES encryption algorithm in CBC mode

The principle of the AES algorithm in CBC mode for encryption implemented in this project is illustrated in the figure 2. It consists of

1. a splitting of the plain text in blocks and creation of the *RoundKey* (figure 10)
2. a *xor* operation
   - between the first block and an initialization vector
   - between the subsequent block and the cipher block created in the previous step
3. the encryption with the *Cipher* function illustrated in figure 3

4. the concatenation of the created cipher blocks

The *Cipher* function starts by an *AddRoundKey* operation (figure 6) followed by a loop of 9 steps. In each step of the loop there are 4 transformations: *SubBytes* (figure 7), *ShiftRows* (figure 8), *MixColumns* (figure 9) and *AddRoundKey*. Afterwards, there is one last step in which we are performing the same operations except *MixColumns*.

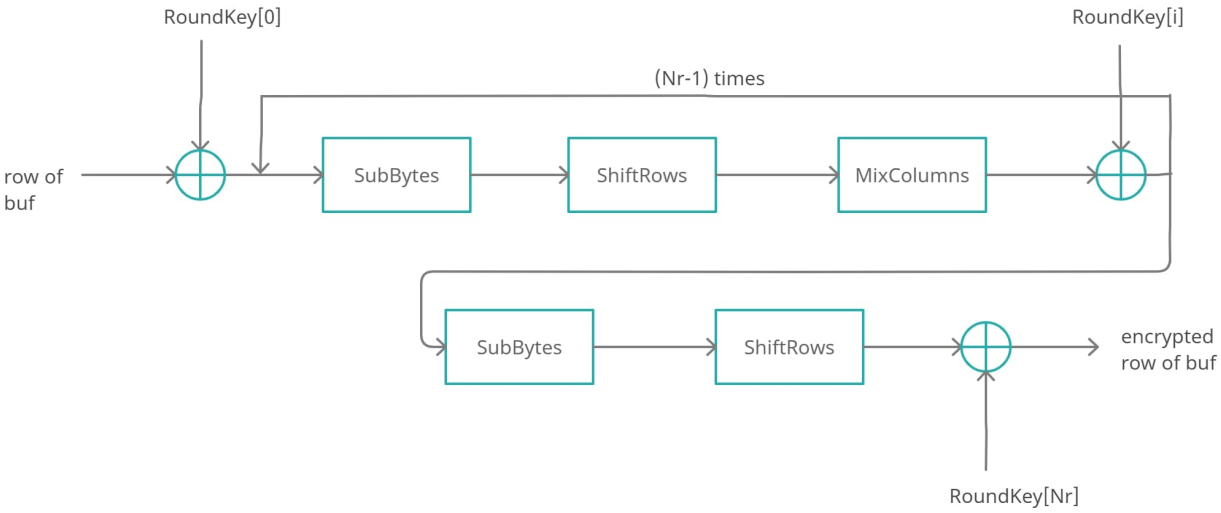

Figure 3: Cipher

3.2 Decryption

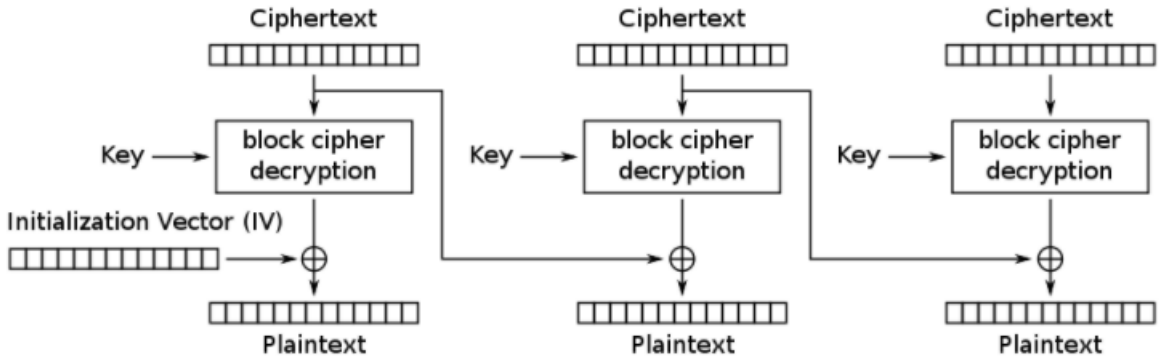

Figure 4: AES decryption algorithm in CBC mode

The principle of the AES algorithm in CBC mode for decryption implemented in this project is illustrated in the figure 4. It consists of

1. a splitting of the cipher text in blocks
2. the decryption with the *InvCipher* function illustrated in figure 5
3. a *xor* operation

- between the first block and an initialization vector
- between the subsequent block and the previous cipher block

4. the concatenation of the created plain blocks to reconstruct the plain text

The *InvCipher* function starts by an *AddRoundKey* operation followed by a loop of 9 steps. In each step of the loop there are 4 transformations: *InvSiftRows* (inverse operation of the above-mentioned *ShiftRows*), *InvSubBytes* (inverse operation of the above-mentioned *SubBytes*), *AddRoundKey* and *InvMixColumns* (inverse operation of the above-mentioned *MixColumns*). Afterwards, there is one last step in which we are performing the same operations except *InvMixColumns*.

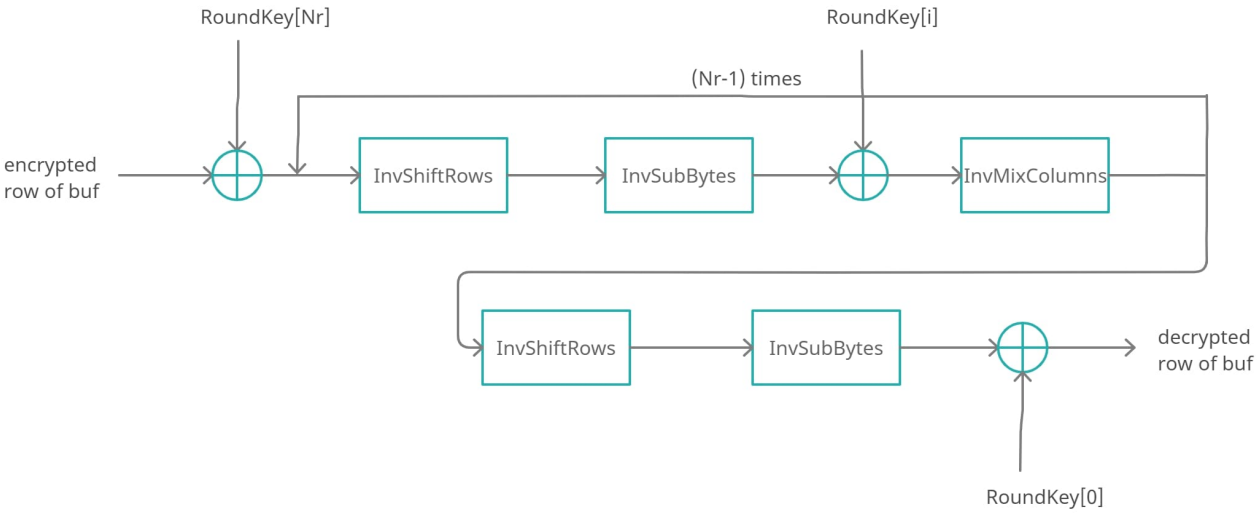

Figure 5: InvCipher

3.3 Simulation

In order to verify the correctness of our AES implementation, we simulated the application using provided µVision simulation tools. Validation of the exactitude of the writing to the desired address (which was set in hardware design defining the BRAM block) and of the accuracy of the encryption/decryption operations was carried out by observing the modification of the memory interface in Keil. Video of the simulation is available at link.

4 Conclusion

The evolution of the project can be briefly described in few phases. First step was installation of the tools needed for development of the project and launch of the DesignStart provided by Arm. We proceeded with the implementation of the simple blinking LED and definition of the final goal of the project. Starting point was exploration and modification of the hardware design and its verification. In order to verify working state of the design, we investigated the simulation tools provided by the software platform and implemented a simple xor encryption. Afterwards, we made a research and implementation of AES algorithm. Finally, we simulated hardware and software simultaneously and obtained expected result.

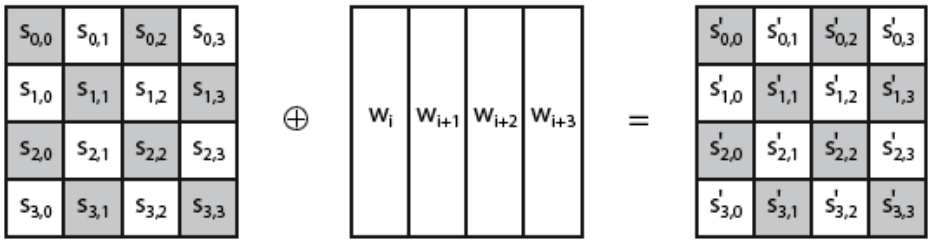

Figure 6: AddRoundKey

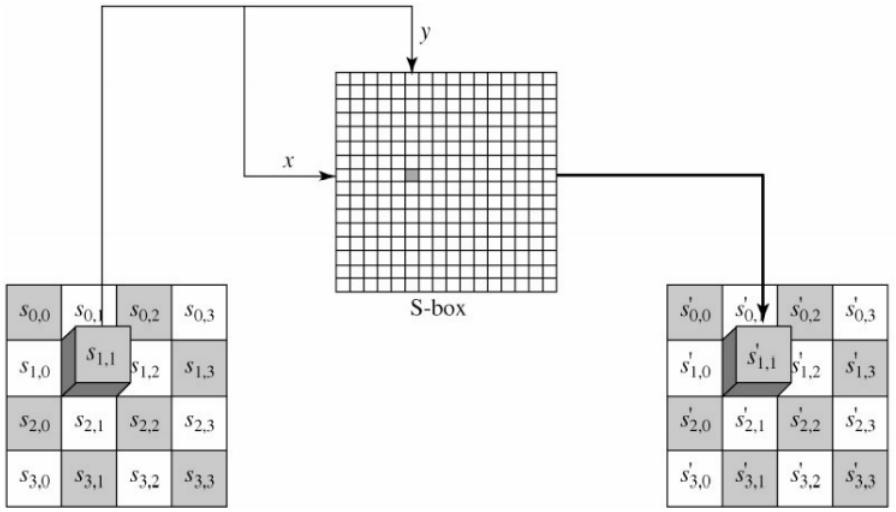

Figure 7: SubBytes

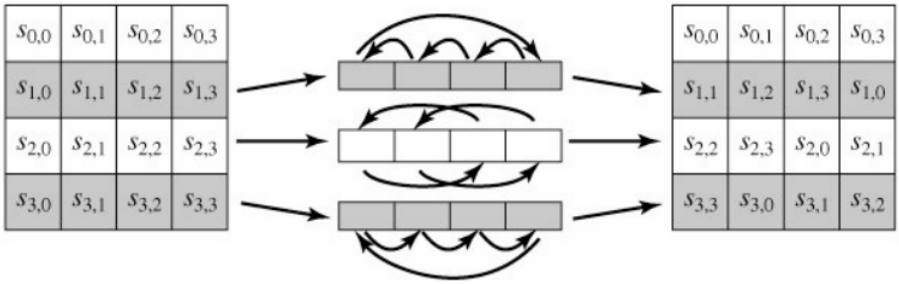

Figure 8: ShiftRows

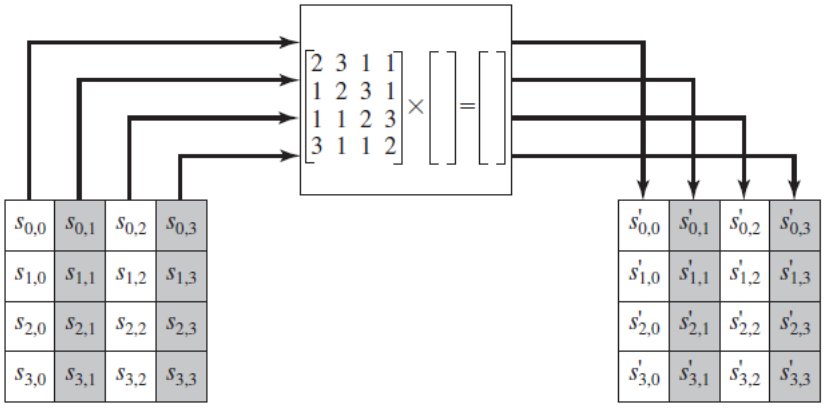

Figure 9: MixColumns

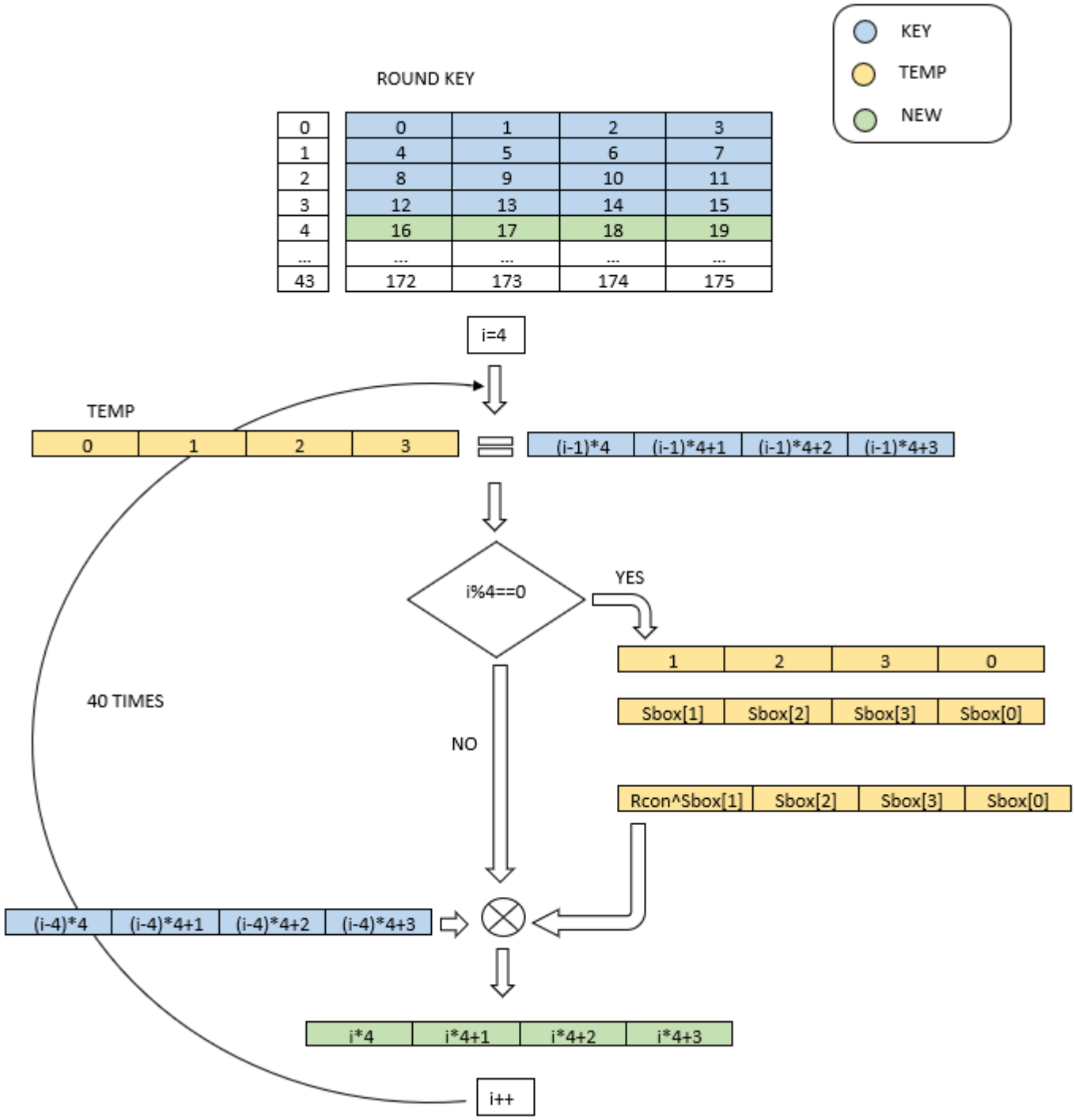

Figure 10: RoundKey