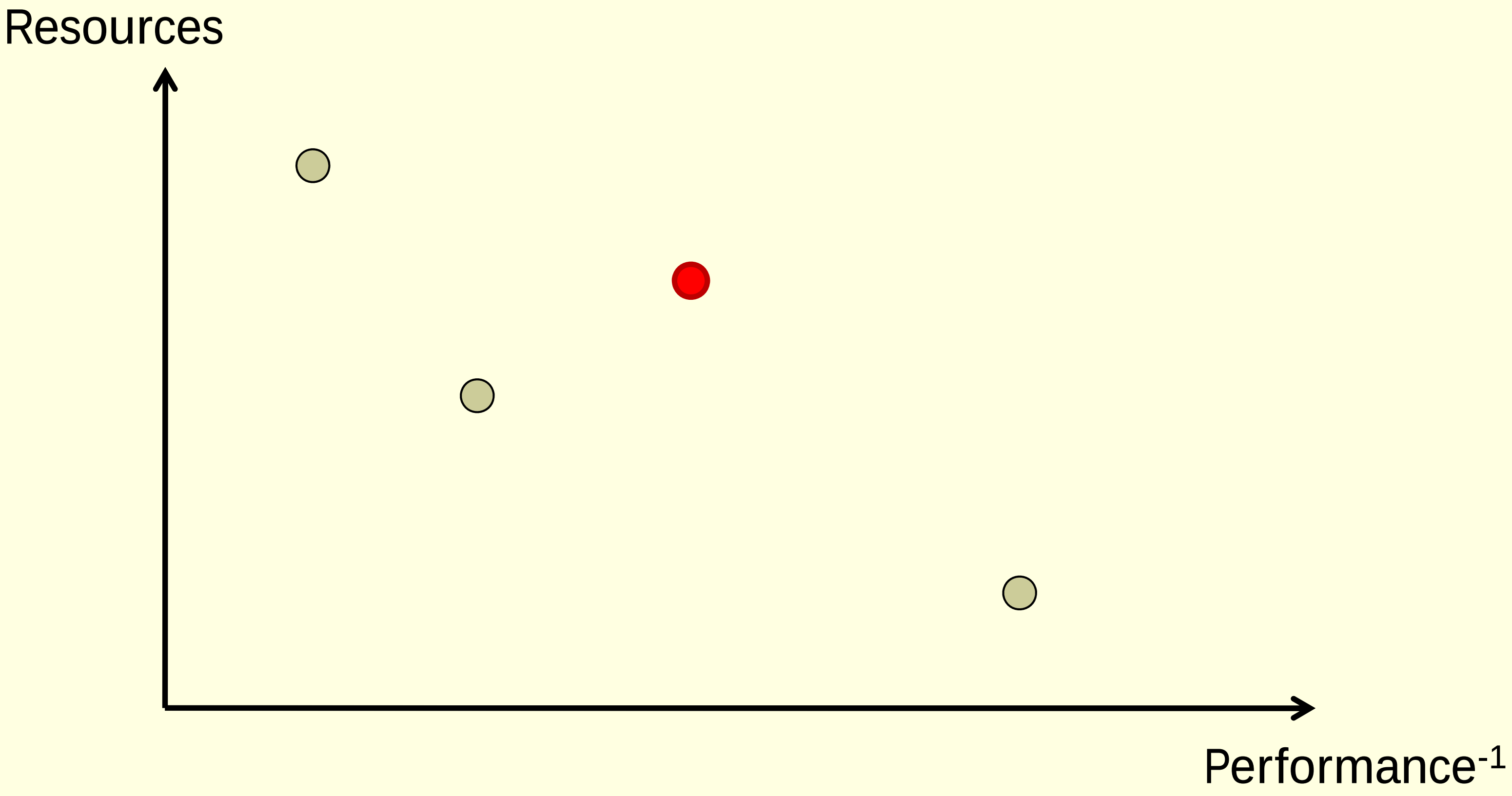
# EE 4218
# AES Major Project Briefing

Akash Kumar

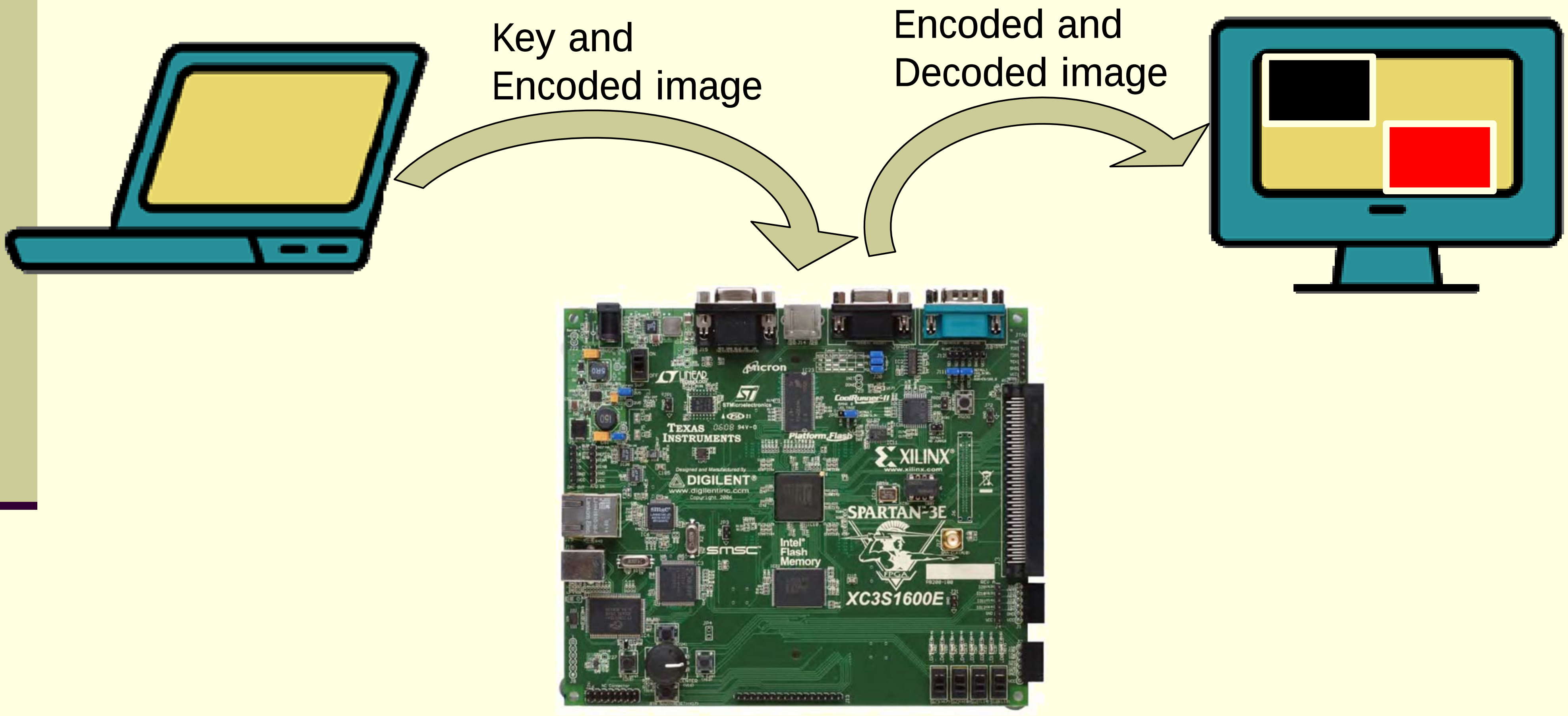
# Project Guidelines

- Work in pairs
- Develop a 128-bit AES decryption engine
  - In C language running on ARM-based platform on FPGA
  - In VHDL running on FPGA
- While one C-implementation is sufficient, multiple VHDL alternatives (at least two) should be explored in terms of area and timing
- HLS (High-level Synthesis) can be a good point
- Measure speed-ups
- Plot all implementations on a Pareto curve

# Pareto Curve



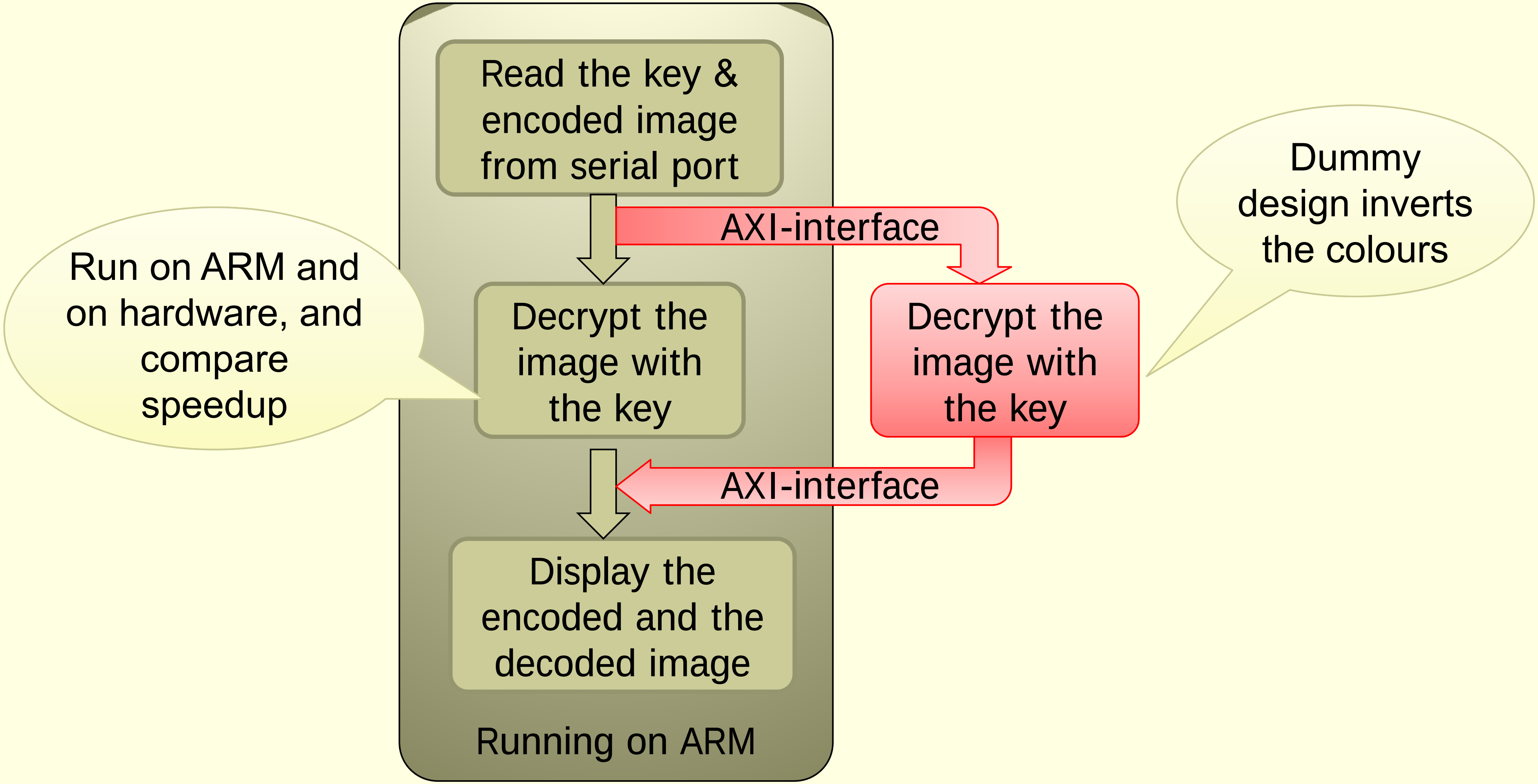
# Project Implementation



Key and
Encoded image

Encoded and
Decoded image

# ARM-based Design


Read the key & encoded image from serial port
AXI-interface
Decrypt the image with the key
Decrypt the image with the key
AXI-interface
Display the encoded and the decoded image
Running on ARM
Run on ARM and on hardware, and compare speedup
Dummy design inverts the colours

# What is Provided!

- A very useful wiki page: http://wiki.nus.edu.sg/display/ee4218
  - Links to resources
  - Updates
  - FAQs
- Use Lab3 as the basis

# ADVANCED ENCRYPTION STANDARD

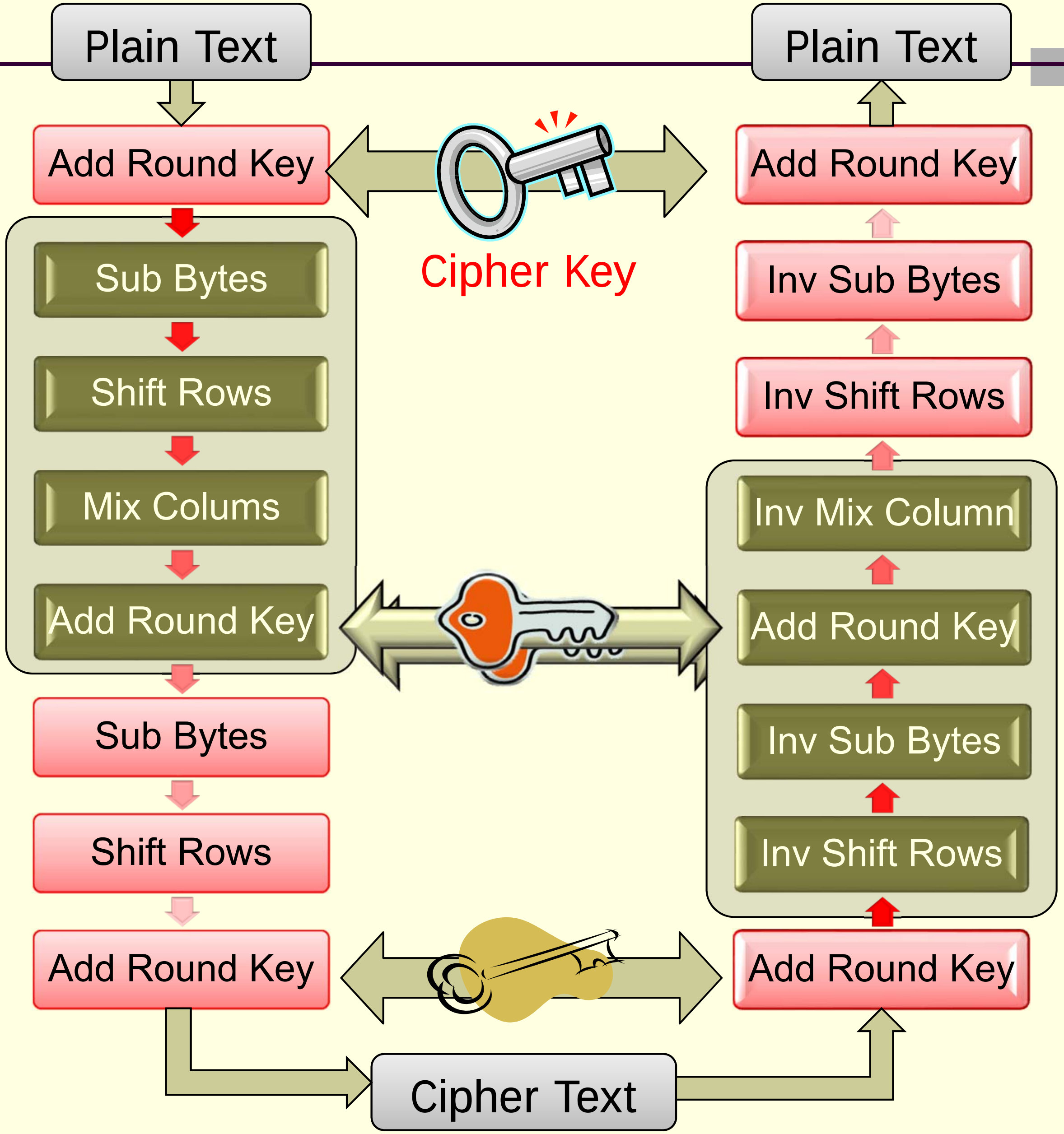
# Sources

- **Cryptography and Network Security** by **Behrouz A. Forouzan**, 2008, (book and slides) Mc Graw Hill.
- AES Wiki page: http://en.wikipedia.org/wiki/Advanced_Encryption_Standard
- Flash animation by Enrique Zabala (very useful): http://www.cs.bc.edu/~straubin/cs381-05/blockciphers/rijndael_ingles2004.swf

# Advanced Encryption Standard (AES)

- AES is a symmetric-key encryption standard adopted by the U.S. government
- Developed by two Belgian cryptographers – Vincent Rijmen and John Daemen: also known as Rijndael
- Standard has three key-sizes – 128, 192, 256 bits
- For the project, use key-size of 128 bits
- Block size is also 128 bits (16 = 4x4 bytes)
- AES ciphers used worldwide.

# Encryption and Decryption



Repeated in
9 Rounds

Repeated in
9 Rounds

# Plain Text (Data) Generation

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $b_0$ | $b_1$ | $b_2$ | $b_3$ | $b_4$ | $b_5$ | $b_6$ | $b_7$ |

1 byte = 8 bits

| | | | |
|---|---|---|---|
| **$b_0$** | **$b_1$** | **$b_2$** | **$b_3$** |

1 Word = 4 bytes

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **$b_0$** | **$b_1$** | **$b_2$** | **$b_3$** | **$b_4$** | **$b_5$** | **$b_6$** | **$b_7$** | **$b_8$** | **$b_9$** | **$b_{10}$** | **$b_{11}$** | **$b_{12}$** | **$b_{13}$** | **$b_{14}$** | **$b_{15}$** |

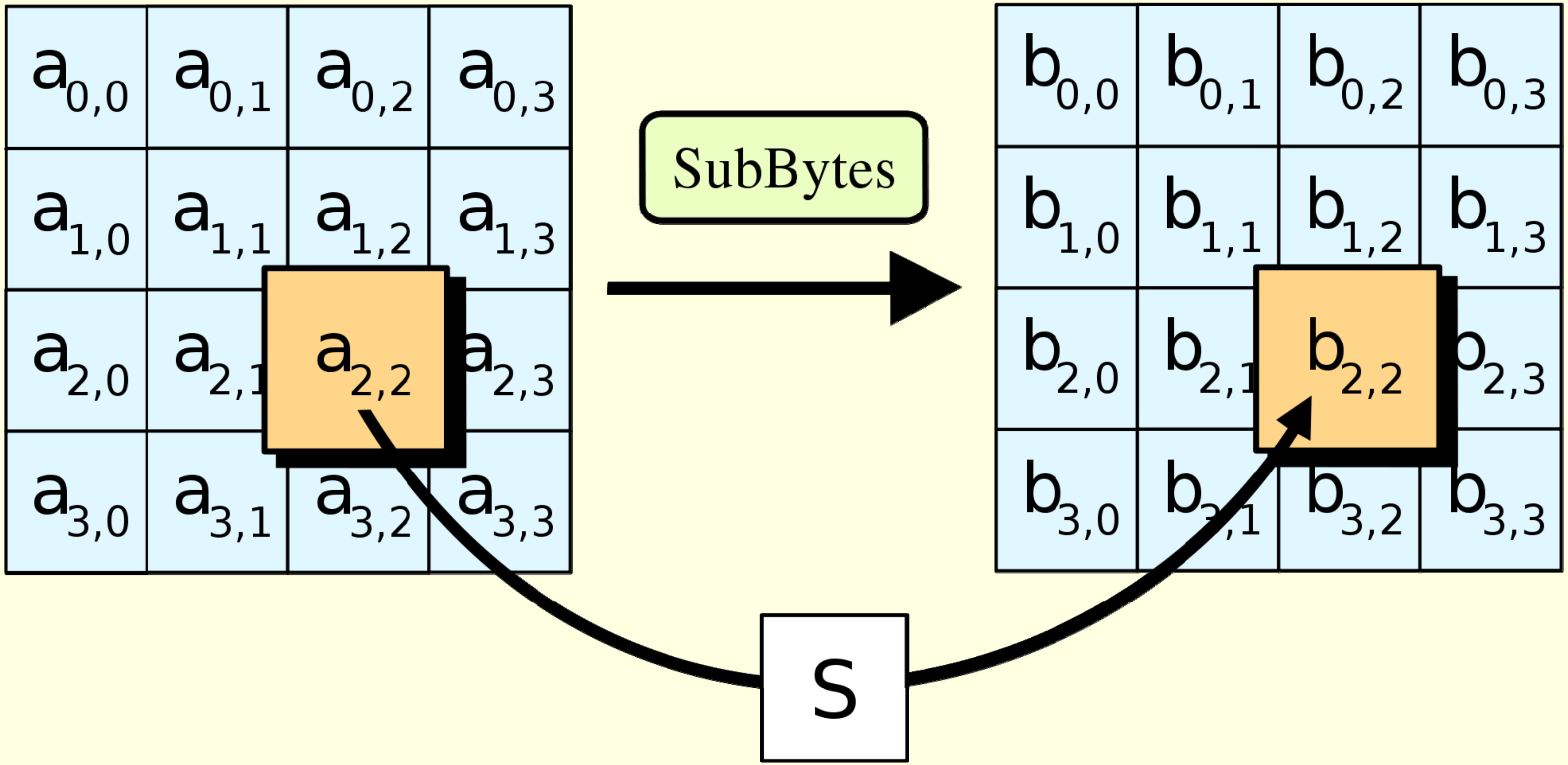
1 Block = 4 words

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

1 State = 1 block

Starting point for AES Encryption

# Substitute Bytes


$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
SubBytes
$b_{0,0}$ $b_{0,1}$ $b_{0,2}$ $b_{0,3}$
$b_{1,0}$ $b_{1,1}$ $b_{1,2}$ $b_{1,3}$
$b_{2,0}$ $b_{2,1}$ $b_{2,2}$ $b_{2,3}$
$b_{3,0}$ $b_{3,1}$ $b_{3,2}$ $b_{3,3}$
S

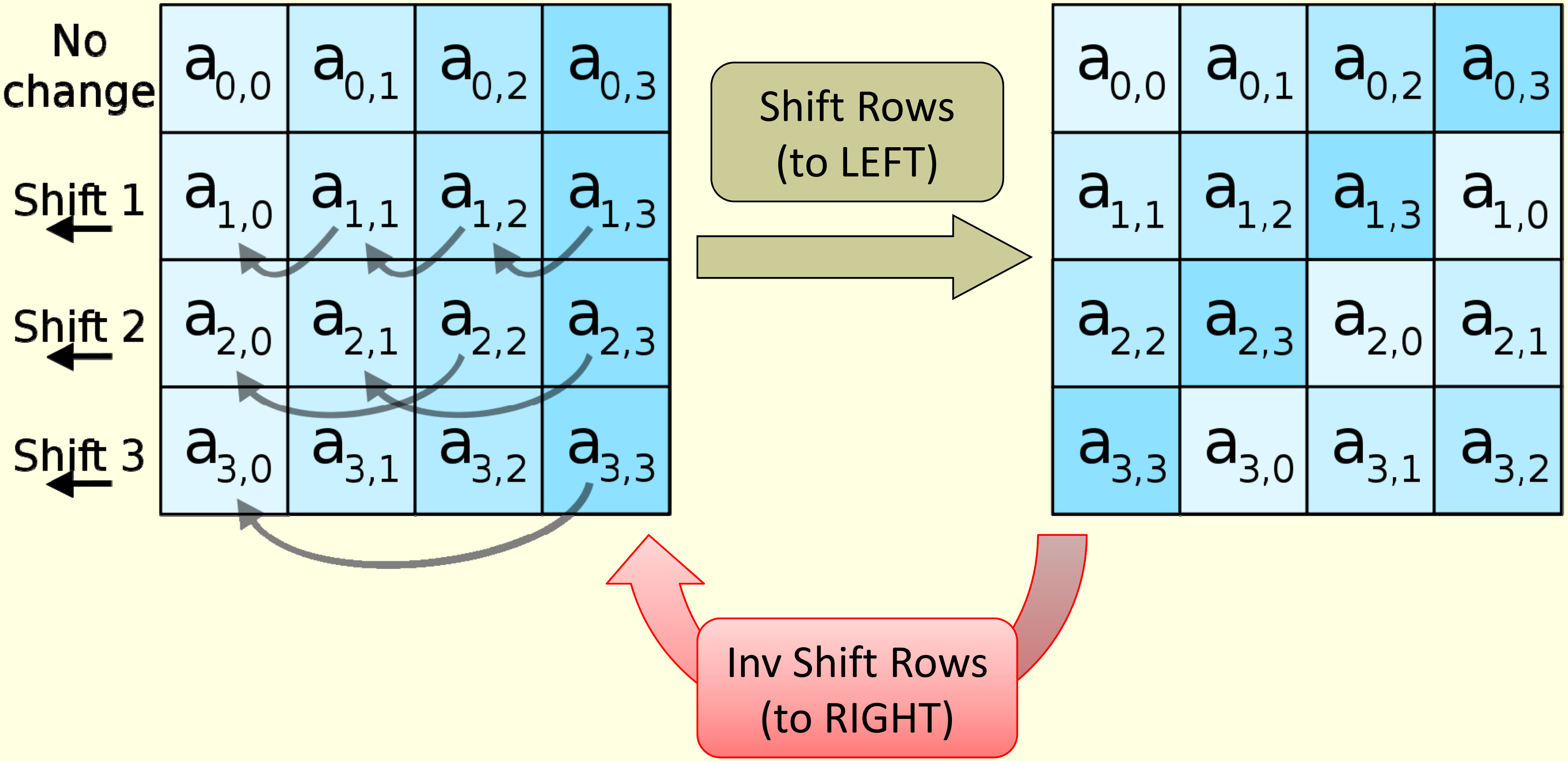
# Substitution Box

| 0x | 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 | 1 | 67 | 2b | fe | d7 | ab | 76 |
| **10** | ca | 82 | c9 | 7d | fa | 59 | 47 | f0 | ad | d4 | a2 | af | 9c | a4 | 72 | c0 |
| **20** | b7 | fd | 93 | 26 | 36 | 3f | f7 | cc | 34 | a5 | e5 | f1 | 71 | d8 | 31 | 15 |
| **30** | 4 | c7 | 23 | c3 | 18 | 96 | 5 | 9a | 7 | 12 | 80 | e2 | eb | 27 | b2 | 75 |
| **40** | 9 | 83 | 2c | 1a | 1b | 6e | 5a | a0 | 52 | 3b | d6 | b3 | 29 | e3 | 2f | 84 |
| **50** | 53 | d1 | 0 | ed | 20 | fc | b1 | 5b | 6a | cb | be | 39 | 4a | 4c | 58 | cf |
| **60** | d0 | ef | aa | fb | 43 | 4d | 33 | 85 | 45 | f9 | 2 | 7f | 50 | 3c | 9f | a8 |
| **70** | 51 | a3 | 40 | 8f | 92 | 9d | 38 | f5 | bc | b6 | da | 21 | 10 | ff | f3 | d2 |
| **80** | cd | 0c | 13 | ec | 5f | 97 | 44 | 17 | c4 | a7 | 7e | 3d | 64 | 5d | 19 | 73 |
| **90** | 60 | 81 | 4f | dc | 22 | 2a | 90 | 88 | 46 | ee | b8 | 14 | de | 5e | 0b | db |
| **a0** | e0 | 32 | 3a | 0a | 49 | 6 | 24 | 5c | c2 | d3 | ac | 62 | 91 | 95 | e4 | 79 |
| **b0** | e7 | c8 | 37 | 6d | 8d | d5 | 4e | a9 | 6c | 56 | f4 | ea | 65 | 7a | ae | 8 |
| **c0** | ba | 78 | 25 | 2e | 1c | a6 | b4 | c6 | e8 | dd | 74 | 1f | 4b | bd | 8b | 8a |
| **d0** | 70 | 3e | b5 | 66 | 48 | 3 | f6 | 0e | 61 | 35 | 57 | b9 | 86 | c1 | 1d | 9e |
| **e0** | e1 | f8 | 98 | 11 | 69 | d9 | 8e | 94 | 9b | 1e | 87 | e9 | ce | 55 | 28 | df |
| **f0** | 8c | a1 | 89 | 0d | bf | e6 | 42 | 68 | 41 | 99 | 2d | 0f | b0 | 54 | bb | 16 |

# Inverse – Substitution Box

| 0x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 52 | 9 | 6a | d5 | 30 | 36 | a5 | 38 | bf | 40 | a3 | 9e | 81 | f3 | d7 | fb |
| 10 | 7c | e3 | 39 | 82 | 9b | 2f | ff | 87 | 34 | 8e | 43 | 44 | c4 | de | e9 | cb |
| 20 | 54 | 7b | 94 | 32 | a6 | c2 | 23 | 3d | ee | 4c | 95 | 0b | 42 | fa | c3 | 4e |
| 30 | 8 | 2e | a1 | 66 | 28 | d9 | 24 | b2 | 76 | 5b | a2 | 49 | 6d | 8b | d1 | 25 |
| 40 | 72 | f8 | f6 | 64 | 86 | 68 | 98 | 16 | d4 | a4 | 5c | cc | 5d | 65 | b6 | 92 |
| 50 | 6c | 70 | 48 | 50 | fd | ed | b9 | da | 5e | 15 | 46 | 57 | a7 | 8d | 9d | 84 |
| 60 | 90 | d8 | ab | 0 | 8c | bc | d3 | 0a | f7 | e4 | 58 | 5 | b8 | b3 | 45 | 6 |
| 70 | d0 | 2c | 1e | 8f | ca | 3f | 0f | 2 | c1 | af | bd | 3 | 1 | 13 | 8a | 6b |
| 80 | 3a | 91 | 11 | 41 | 4f | 67 | dc | ea | 97 | f2 | cf | ce | f0 | b4 | e6 | 73 |
| 90 | 96 | ac | 74 | 22 | e7 | ad | 35 | 85 | e2 | f9 | 37 | e8 | 1c | 75 | df | 6e |
| a0 | 47 | f1 | 1a | 71 | 1d | 29 | c5 | 89 | 6f | b7 | 62 | 0e | aa | 18 | be | 1b |
| b0 | fc | 56 | 3e | 4b | c6 | d2 | 79 | 20 | 9a | db | c0 | fe | 78 | cd | 5a | f4 |
| c0 | 1f | dd | a8 | 33 | 88 | 7 | c7 | 31 | b1 | 12 | 10 | 59 | 27 | 80 | ec | 5f |
| d0 | 60 | 51 | 7f | a9 | 19 | b5 | 4a | 0d | 2d | e5 | 7a | 9f | 93 | c9 | 9c | ef |
| e0 | a0 | e0 | 3b | 4d | ae | 2a | f5 | b0 | c8 | eb | bb | 3c | 83 | 53 | 99 | 61 |
| f0 | 17 | 2b | 4 | 7e | ba | 77 | d6 | 26 | e1 | 69 | 14 | 63 | 55 | 21 | 0c | 7d |

# Shift Rows/ Inverse Shift Rows



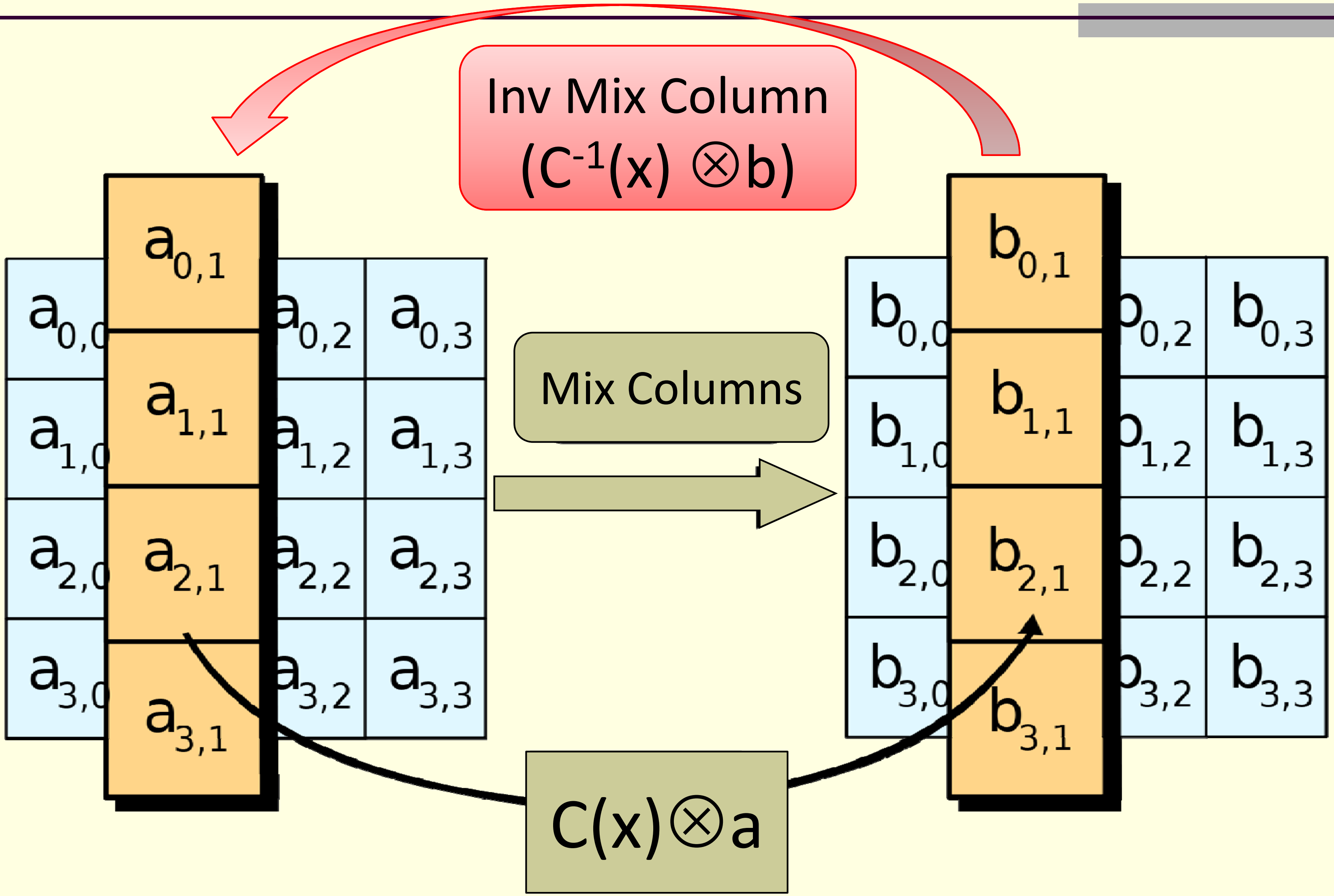
|  |  |  |  |  |
|---|---|---|---|---|
| No change | $a_{0,0}$ | $a_{0,1}$ | $a_{0,2}$ | $a_{0,3}$ |
| Shift 1 ← | $a_{1,0}$ | $a_{1,1}$ | $a_{1,2}$ | $a_{1,3}$ |
| Shift 2 ← | $a_{2,0}$ | $a_{2,1}$ | $a_{2,2}$ | $a_{2,3}$ |
| Shift 3 ← | $a_{3,0}$ | $a_{3,1}$ | $a_{3,2}$ | $a_{3,3}$ |

Shift Rows
(to LEFT)

|  |  |  |  |
|---|---|---|---|
| $a_{0,0}$ | $a_{0,1}$ | $a_{0,2}$ | $a_{0,3}$ |
| $a_{1,1}$ | $a_{1,2}$ | $a_{1,3}$ | $a_{1,0}$ |
| $a_{2,2}$ | $a_{2,3}$ | $a_{2,0}$ | $a_{2,1}$ |
| $a_{3,3}$ | $a_{3,0}$ | $a_{3,1}$ | $a_{3,2}$ |

Inv Shift Rows
(to RIGHT)

# Mix Columns/ Inverse Mix Columns


Inv Mix Column
$(C^{-1}(x) \otimes b)$
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
Mix Columns
$b_{0,0}$ $b_{0,1}$ $b_{0,2}$ $b_{0,3}$
$b_{1,0}$ $b_{1,1}$ $b_{1,2}$ $b_{1,3}$
$b_{2,0}$ $b_{2,1}$ $b_{2,2}$ $b_{2,3}$
$b_{3,0}$ $b_{3,1}$ $b_{3,2}$ $b_{3,3}$
$C(x) \otimes a$

# Mix Columns/ Inverse Mix Columns

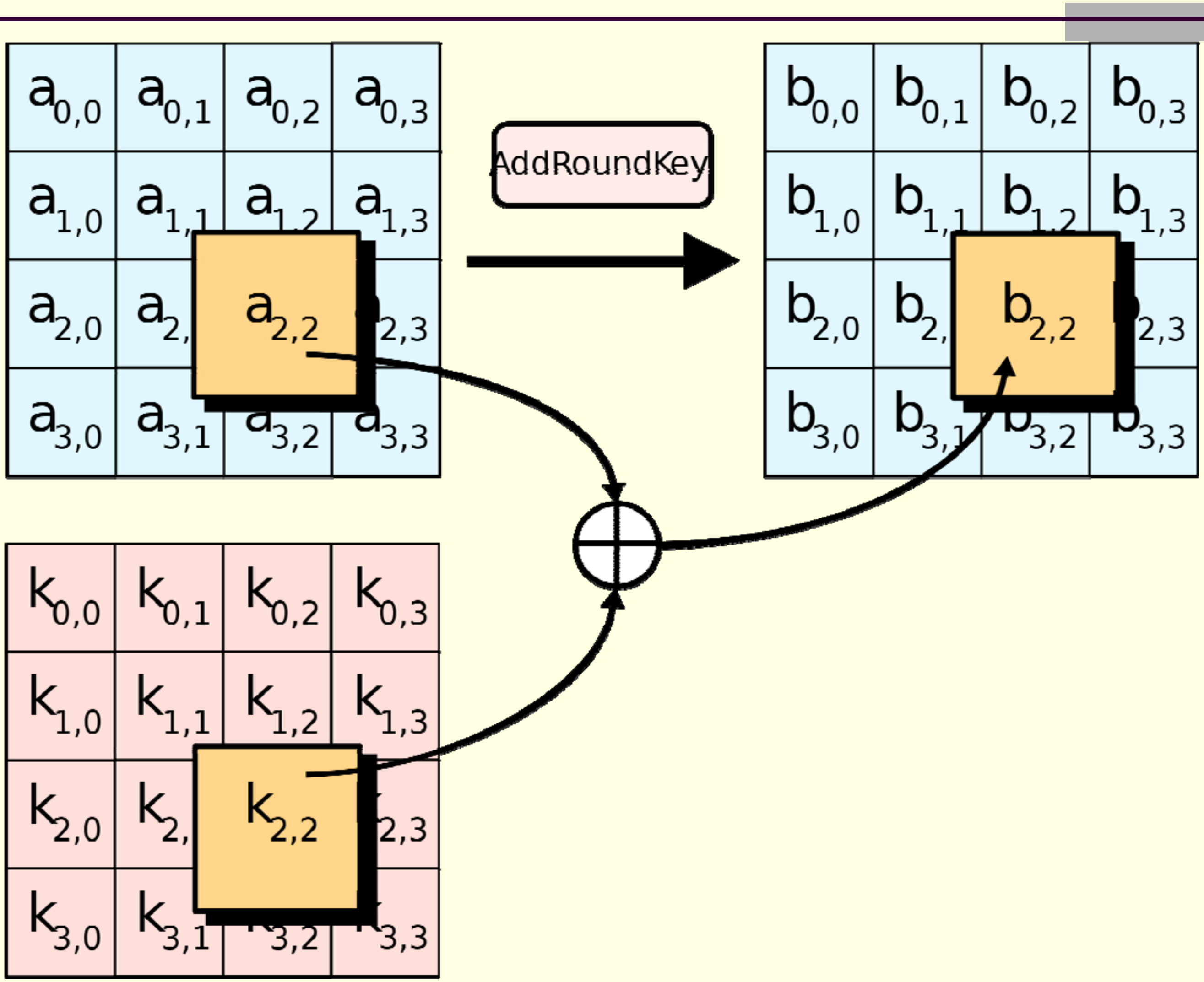
$$B = C \times A = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix} = \begin{bmatrix} 2x + 3y + z + t \\ x + 2y + 3z + t \\ x + y + 2z + 3t \\ 3x + y + z + 2t \end{bmatrix}$$

$$C^{-1} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix}$$

$$C^{-1} \times B = C^{-1} \times C \times A = A$$

Inverse Mix Column

# Add Round Keys


$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
AddRoundKey
$b_{0,0}$ $b_{0,1}$ $b_{0,2}$ $b_{0,3}$
$b_{1,0}$ $b_{1,1}$ $b_{1,2}$ $b_{1,3}$
$b_{2,0}$ $b_{2,1}$ $b_{2,2}$ $b_{2,3}$
$b_{3,0}$ $b_{3,1}$ $b_{3,2}$ $b_{3,3}$
$k_{0,0}$ $k_{0,1}$ $k_{0,2}$ $k_{0,3}$
$k_{1,0}$ $k_{1,1}$ $k_{1,2}$ $k_{1,3}$
$k_{2,0}$ $k_{2,1}$ $k_{2,2}$ $k_{2,3}$
$k_{3,0}$ $k_{3,1}$ $k_{3,2}$ $k_{3,3}$

# Add Round Keys

- Bit-wise add is essentially an XOR operation
- XOR operation

| A | B | Z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

$$A \oplus K = B$$

$$B \oplus K = A \oplus K \oplus K = A \oplus 0 = A$$

- Therefore, adding round key two times returns original value (so long as the key is the same)

# KEY EXPANSION

# Key Expansion

- 11 different keys needed in total
- First one is the cipher key
  - Other ten generated from cipher key
- 128-bit AES = 16-byte AES i.e. 16 byte KEY

| $b_0$ | $b_4$ | $b_8$ | $b_{12}$ |
|---|---|---|---|
| $b_1$ | $b_5$ | $b_9$ | $b_{13}$ |
| $b_2$ | $b_6$ | $b_{10}$ | $b_{14}$ |
| $b_3$ | $b_7$ | $b_{11}$ | $b_{15}$ |

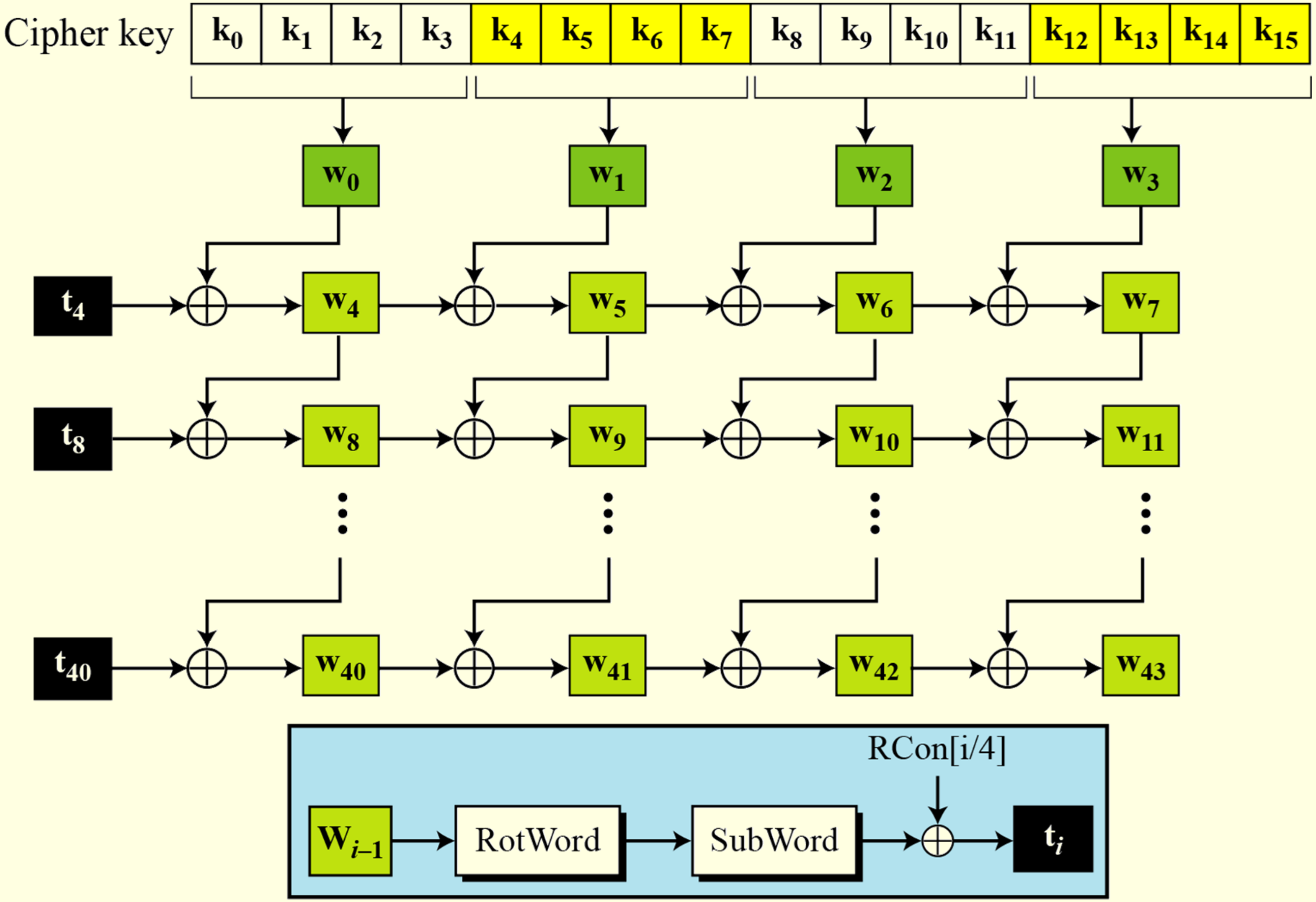
=

| $W_0$ | $W_1$ | $W_2$ | $W_3$ |
|---|---|---|---|

# Key Expansion

| Round | Key |
|---|---|
| Pre-round | w0 w1 w2 w3 |
| Round 1 | w4 w5 w6 w7 |
| Round 2 | … |
| Round 3 | … |
| Round 4 | … |
| Round 5 | … |
| Round 6 | … |
| Round 7 | … |
| Round 8 | … |
| Round 9 | … |
| Round 10 | w40 w41 w42 w43 |

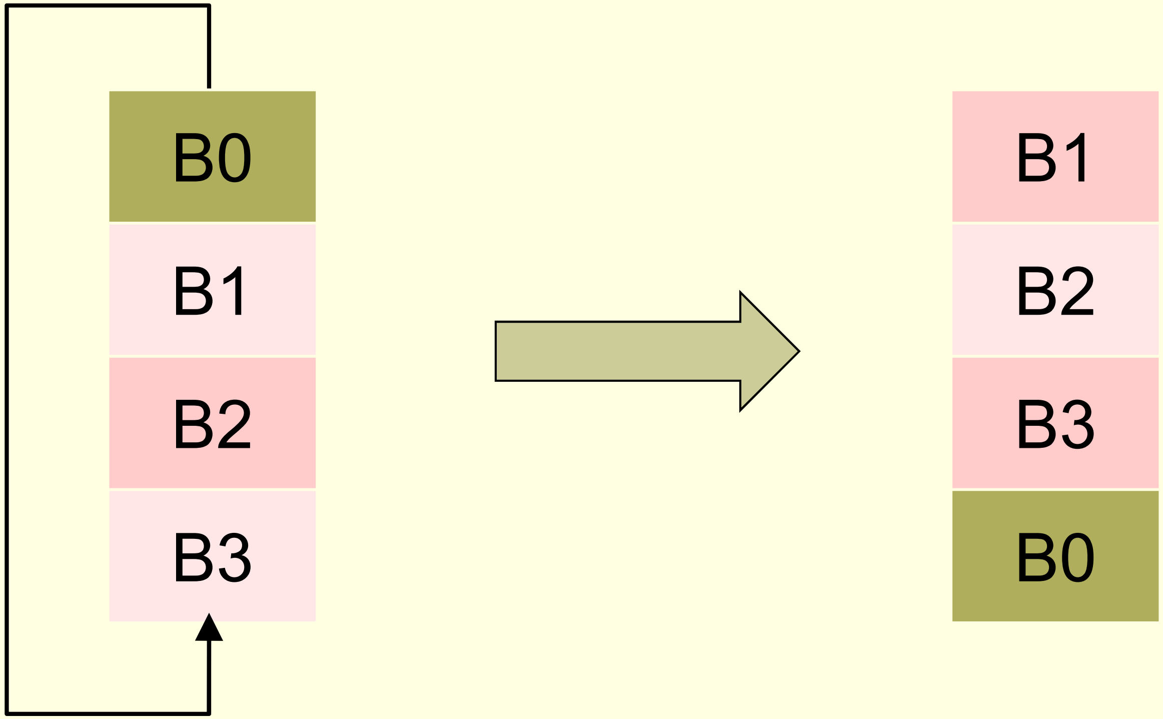
# Key Expansion


Cipher key
$k_0$ $k_1$ $k_2$ $k_3$ $k_4$ $k_5$ $k_6$ $k_7$ $k_8$ $k_9$ $k_{10}$ $k_{11}$ $k_{12}$ $k_{13}$ $k_{14}$ $k_{15}$
$w_0$ $w_1$ $w_2$ $w_3$
$t_4$ $w_4$ $w_5$ $w_6$ $w_7$
$t_8$ $w_8$ $w_9$ $w_{10}$ $w_{11}$
$t_{40}$ $w_{40}$ $w_{41}$ $w_{42}$ $w_{43}$
RCon[i/4]
$W_{i-1}$
RotWord
SubWord
$t_i$

Making of $t_i$ (temporary) words $i = 4\ N_r$

# Rotate Word & Substitute Word

- RotWord: Rotates the four bytes in a word by one byte



- SubWord: substitute individual bytes by subByte

# Round Constants

- In each round a different word is added (XOR'ed)

| R1 | R2 | R3 | R4 | R5 | R6 | R7 | R8 | R9 | R10 |
|---|---|---|---|---|---|---|---|---|---|
| 01 | 02 | 04 | 08 | 10 | 20 | 40 | 80 | 1B | 36 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |

# Observations

- Note that for decryption the keys are needed in reverse order
- Key schedule generation can be pipelined – note the dependencies in Slide 23
- See if there is any potential of hardware reuse
- You may want to test your design with test vectors provided in official AES standard – available on the EE4218 Project Wiki