# 05. Optimalizace AES pro 32bitovou platformu

1. Změřte výkonnost své implementace AES z předchozího cvičení
    - vytvořte novou verzi zdrojového kódu,
    - odstraňte všechny výpisy, které by mohly zdržovat výpočet
    - funkce word a wbyte inlinujte nebo přepište na makra
    - vytvořte cyklus, ve kterém budete volat šifru AES opakovaně
    - volejte AES v režimu OFB, tedy vstup i výstup bude stejné pole, abyste vytvořili datovou závislost mezi jednotlivými voláními
    - počet opakování bude nepovinný parametr programu na příazové řádce, výchozí hodnota bude 1000000
    - v programu měřte celkový čas potřebný pro provedení smyčky. Pro měření použijte funkce časovače s vysokým rozlišením, pokud to platforma podporuje (např. `std::chrono` , reference (https://en.cppreference.com/w/cpp/chrono) )
    - výsledný čas vypište na standardní výstup v milisekundách.
    - návratový kód programu nastavte na hodnotu výstupního bajtu out[0], abyste použili výstup a nedali překladači příležitost k odstranění kódu
    - program přeložte s optimalizacemi (-Ofast v gcc, Release profil v MSVS)
2. Optimalizujte AES pomocí T-boxů
    - vytvořte novou verzi zdrojového kódu
    - před prvním šifrováním AES vygenerujte tabulky T0 až T3
    - v hlavní smyčce AES nahraďte trojici SubBytes, ShiftRows, MixColumns s použitím tabulek
    - otestujte správnost výpočtu, nalezené chyby opravte
    - otestujte správnost výpočtu, nalezené chyby opravte
    - otestujte správnost výpočtu, nalezené chyby opravte
    - změřte výkonnost jako výše
3. Porovnejte své implementace a výsledky zapište do tabulky
    - nahrajte své zdrojové kódy + makefile, aby se dal celý projekt hned přeložit a otestovat – `make` pro překlad, `make run` pro spuštění
        a. Původní AES z minulého cvičení
        b. Nový program měřící výkonnost neoptimalizované verze
        c. Nový program s T-boxy
        d. Tabulka bude obsahovat následující sloupce
            - Název varianty (název souboru se zdrojovým kódem)
            - Čas pro 1 milion iterací
            - Čas pro 10 milionů iterací

- Typ procesoru
- Typ OS
- Typ překladače
- Přepínače překladače

## T-boxes

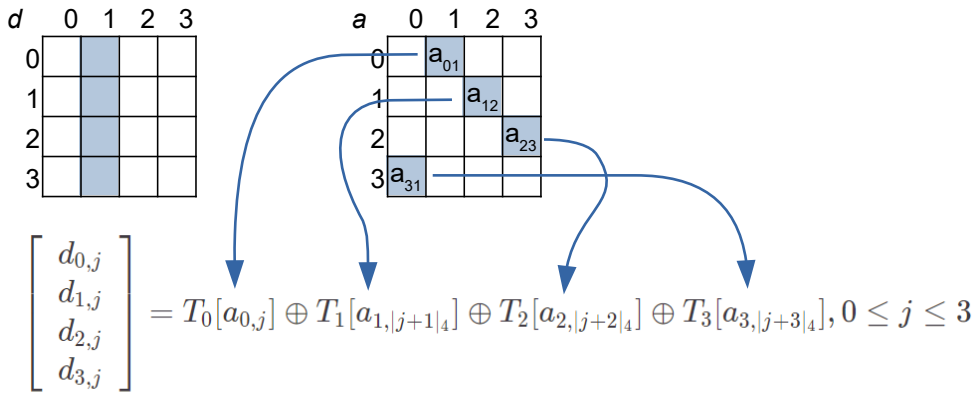
- Předem spočítáme obsah 4 tabulek T0 až T3 pro všechny možné hodnoty bajtu *a*:

$$T_0[a] = \begin{bmatrix} 02 \cdot S(a) \\ 01 \cdot S(a) \\ 01 \cdot S(a) \\ 03 \cdot S(a) \end{bmatrix}, T_1[a] = \begin{bmatrix} 03 \cdot S(a) \\ 02 \cdot S(a) \\ 01 \cdot S(a) \\ 01 \cdot S(a) \end{bmatrix}, T_2[a] = \begin{bmatrix} 01 \cdot S(a) \\ 03 \cdot S(a) \\ 02 \cdot S(a) \\ 01 \cdot S(a) \end{bmatrix}, T_3[a] = \begin{bmatrix} 01 \cdot S(a) \\ 01 \cdot S(a) \\ 03 \cdot S(a) \\ 02 \cdot S(a) \end{bmatrix}.$$

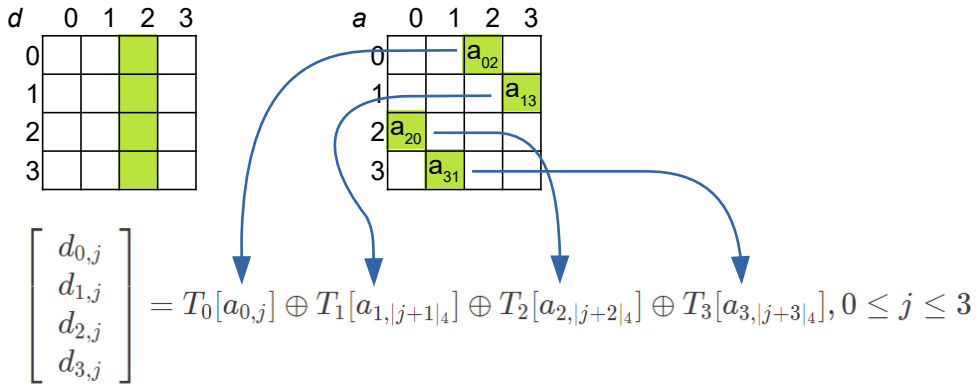
- Každý sloupec po MixColumns ($d_{i,j}$) lze spočítat ze vstupu rundy ($a_{i,j}$) jako XOR-suma čtyř položek z tabulek:

$$\begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} = T_0[a_{0,j}] \oplus T_1[a_{1,|j+1|_4}] \oplus T_2[a_{2,|j+2|_4}] \oplus T_3[a_{3,|j+3|_4}], 0 \leq j \leq 3$$

### Příklad pro j=1




### Příklad pro j=2




$$\begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} = T_0[a_{0,j}] \oplus T_1[a_{1,|j+1|_4}] \oplus T_2[a_{2,|j+2|_4}] \oplus T_3[a_{3,|j+3|_4}], 0 \leq j \leq 3$$