**História de MixColumns**

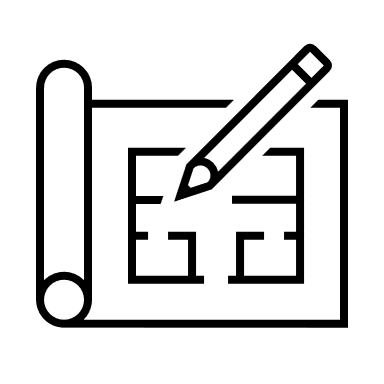
| **Épico** |  |
| --- | --- |
| **Feature** |  |
| **História** | **MixColumns** |

Modelo 3W (Who, What, Why?)

**Quem?**

**Eu quero:** Implementar e validar o módulo MixColumns, seguindo o algoritmo AES-128, e o padrão FIPS-197.

**Para que***:* Garantir e difusão dos dados durante a criptografia. Assegurar que cada byte de saída dependa de todos os bytes da coluna de entrada

**Detalhamento de negócios**

(Coloque aqui de forma resumida e clara o que precisa ser considerado na parte de negócios, o que a história faz, regras de negócio)

O MixColumns é responsável por misturar os bytes de cada coluna da matriz de estado do AES durante a criptografia, promovendo difusão dos dados cifrados e dificultando ataques ao texto cifrado.

Regras a considerar:

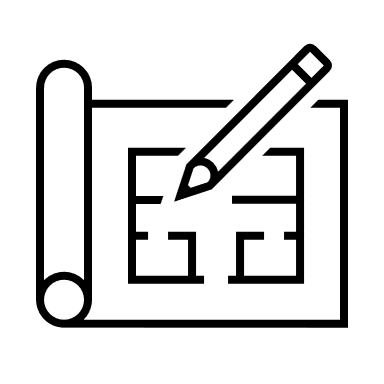
* A transformação deve seguir exatamente o padrão definido pelo AES, utilizando a matriz de multiplicação correta e operações no corpo finito GF(2^8)
* Garantir que cada byte de saída dependa de todos os bytes da coluna de entrada.

<https://repositorio.ufpb.br/jspui/bitstream/tede/6092/1/arquivototal.pdf>

<https://crypto.stackexchange.com/questions/2402/how-to-solve-mixcolumns/95775#95775>

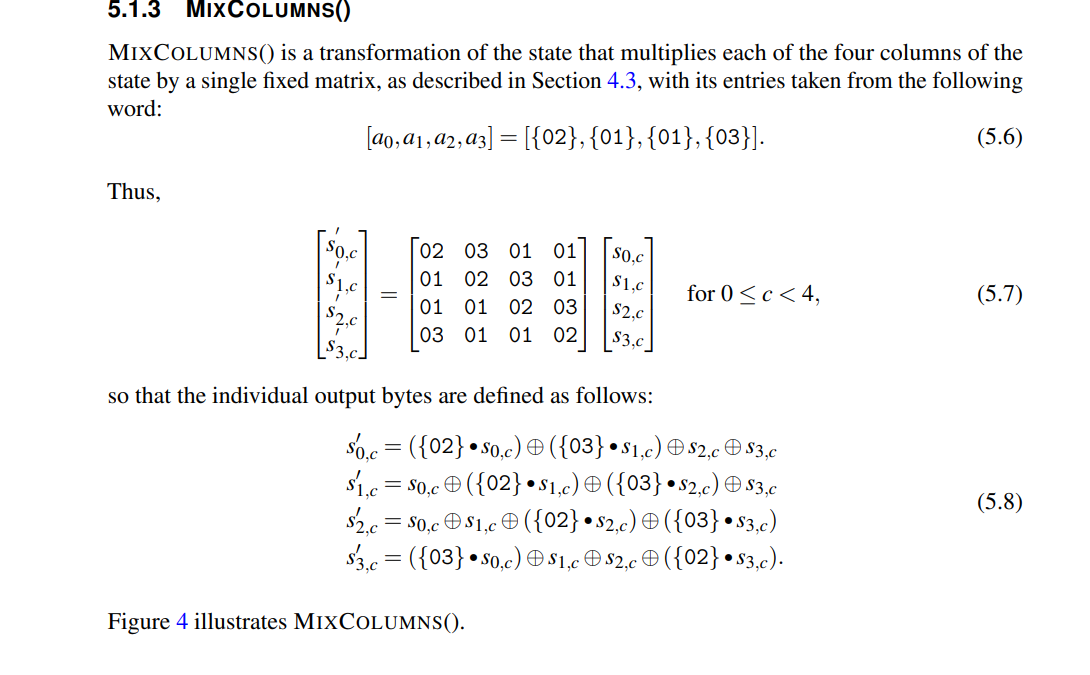
<http://wiki.stoa.usp.br/images/1/10/Cap5-TES.pdf>

https://www.lncc.br/~borges/doc/Aplica%C3%A7%C3%B5es%20de%20GF(2%5E8)%20no%20algoritmo%20AES.pdf

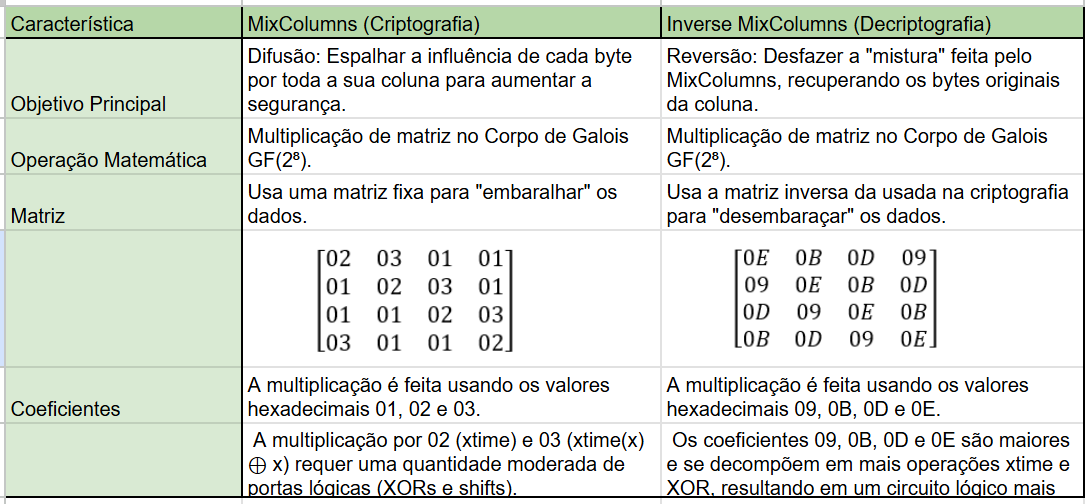
**Detalhamento técnico**

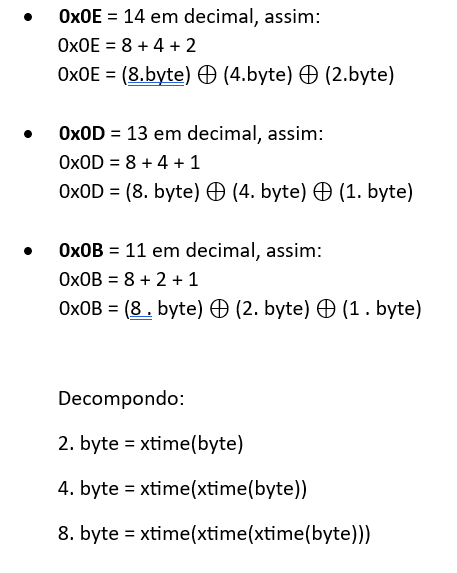
(Coloque aqui de forma resumida o que precisa ser considerado na parte técnica para o desenvolvimento da história)

Realiza uma multiplicação matricial entre cada coluna da matriz de estado (state) e uma matriz fixa 4x4, utilizando operações no corpo finito GF(2⁸). Cada byte da coluna é multiplicado por um coeficiente específico da matriz, e os resultados são somados (usando XOR).

**

* Utilizar funções para multiplicação no corpo finito GF(2^8)
* Aplicar a matriz padrão do MixColumns a cada coluna da matriz de estado.
* Garantir que a lógica seja compatível com o fluxo de dados do AES e com as demais etapas do algoritmo.
* Prever a operação inversa (Inverse MixColumns) para o processo de decriptação.

****

****

**Exemplos**

<https://git.uni-paderborn.de/qazi/malicious-routing/-/blob/master/Examples/aes_example/verilog/mixcolumn.v>

Função auxiliar para coluna:

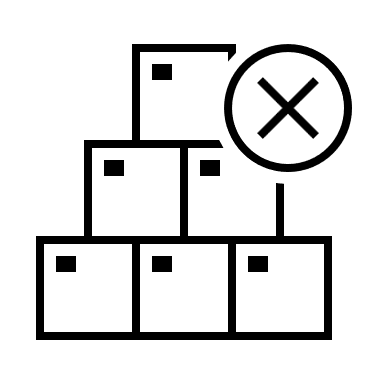
[*https://github.com/pnvamshi/Hardware-Implementation-of-AES-Verilog/blob/master/AES-128-Bit-Verilog/mixcolumn.v*](https://github.com/pnvamshi/Hardware-Implementation-of-AES-Verilog/blob/master/AES-128-Bit-Verilog/mixcolumn.v)

*COM TESTBENCH:*

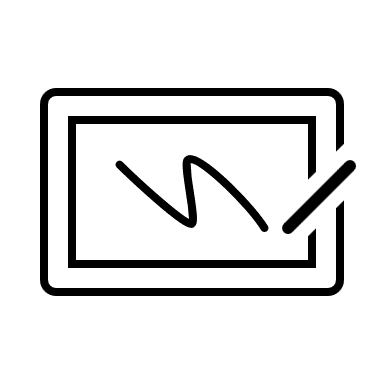
[*https://github.com/pr-mittal/AES128\_Verilog/blob/master/general/t\_mix\_columns.v*](https://github.com/pr-mittal/AES128_Verilog/blob/master/general/t_mix_columns.v)

<https://github.com/michaelehab/AES-Verilog/blob/main/mixColumns.v>

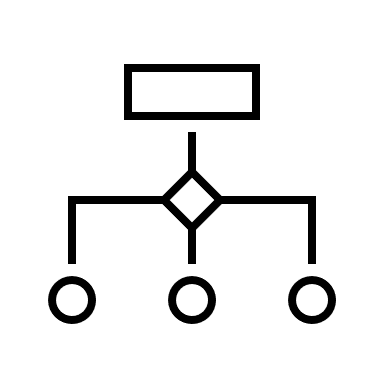
*https://www.lncc.br/~borges/doc/Aplica%C3%A7%C3%B5es%20de%20GF(2%5E8)%20no%20algoritmo%20AES.pdf*

**Dependências/Premissas**

(Coloque aqui as premissas ou dependências da história)

**Saídas**

| Função | Nome | Descrição |
| --- | --- | --- |
| Entrada da Coluna |  |  |
| Saída da Coluna |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Fluxo TOBE e refinamento técnico**

(Coloque aqui o fluxo TOBE para essa história e/ou fluxo da solução)

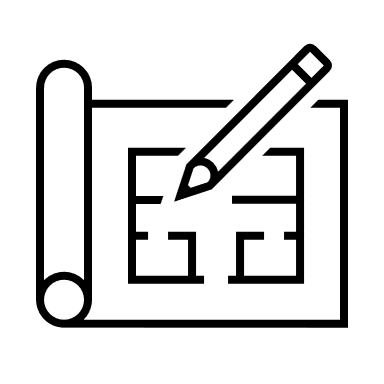
1. Receber a matriz de estado AES.

2. Para cada coluna:

Aplicar multiplicação matricial com a matriz padrão

Gerar nova coluna resultante

3. Passar o resultado para a próxima etapa do algoritmo AES

**Casos de teste**

**Cenário 1:**

| Item | Descrição |
| --- | --- |
| Nome do Teste |  |
| Objetivo |  |
| Ambiente |  |
| Procedimento |  |
| Resultado Esperado |  |
| Registro dos Resultados |  |
| Conclusão |  |

**Cenário 2:**

| Item | Descrição |
| --- | --- |
| Nome do Teste |  |
| Objetivo |  |
| Ambiente |  |
| Procedimento |  |
| Resultado Esperado |  |
| Registro dos Resultados |  |
| Conclusão |  |

**DoR**



**Critérios de aceite**

(Coloque aqui os critérios de aceite)

| Critério DoR | Check |
| --- | --- |
| 1. Dependências sanadas ou mapeadas; |  |
| 2. Preparação dos ambientes (equalização); |  |
| 3. Acessos e perfis liberados; |  |
| 4. Critérios de aceite definidos e aprovados; |  |
| 5. História compreendida e alinhada pelo time; |  |
| 6. Estimativa completa da equipe de desenvolvimento; |  |
| 7. Definição de cenários de teste e validação; |  |

DoR (Definition of Ready): itens que garatem que a equipe está preparada para começar a trabalhar a história.

**DoD**



| Critério | Check |
| --- | --- |
| 1. Passou por revisão de código (boas práticas, threads, etc); |  |
| 2. Atende os critérios de aceite. |  |
| 3. Cenários testados. |  |
| 4. Disponível para implantação em ambiente de homologação. |  |
| 5. Histórias homologadas pelo PO; |  |
| 6. Demonstrado para as partes interessadas; |  |
| 7. Documentação técnica de entrega; |  |
| 8. Implantação em produção; |  |
|  |  |
|  |  |

\*\*DoD (Definition of Done): itens que garantem que a história está completa e atende a todos os requisitos.