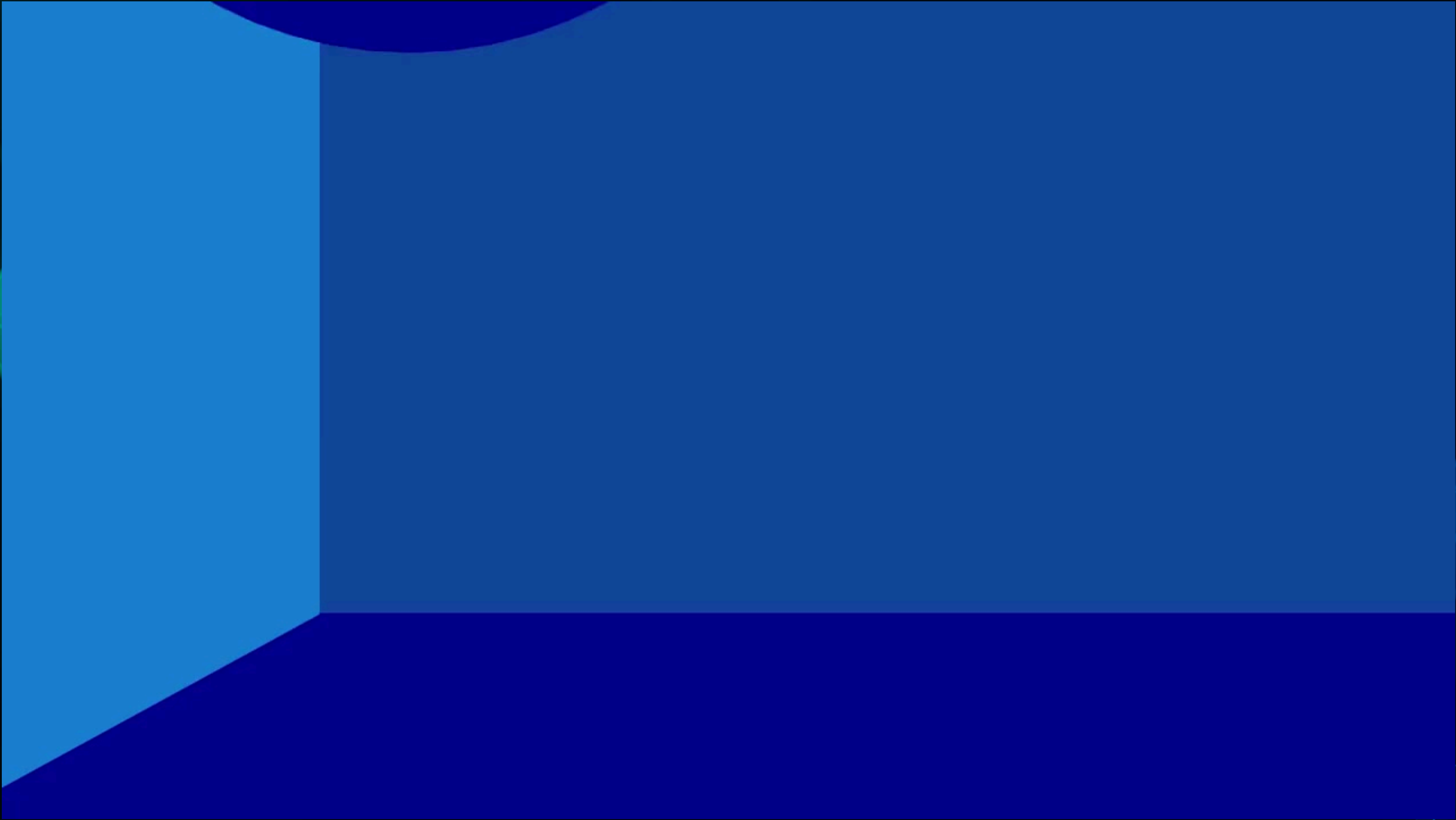


PROYECTO DE ENCRYPTACIÓN Y DESENCRIPTACIÓN EN HARDWARE

Implementación del algoritmo AES

TE2002B.401



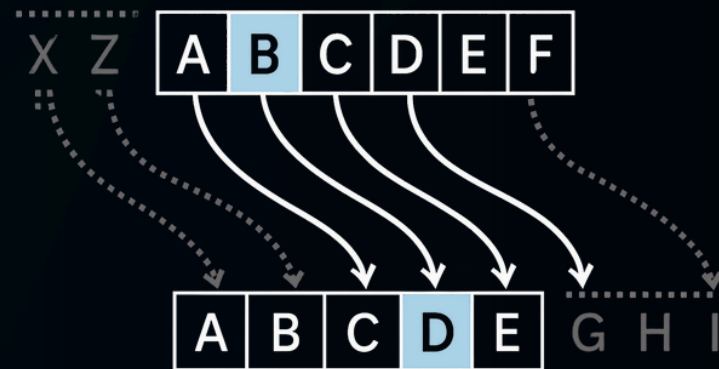
¿QUÉ ES LA CRIPTOGRAFÍA?

Técnica para ocultar o proteger información usando claves. Ha evolucionado desde simples desplazamientos hasta algoritmos matemáticos complejos.



ORÍGENES

CÉSAR



VIGENÉRE

Mensaje: ATAQUE

Clave: SOL

Repetimos la clave:

Mensaje: A T A Q U E
Clave: S O L S O L

ENIGMA (Alan Turing)



DES

(Cifrado electrónico)



*Advanced Encryption
Standard (2001) (Rijndael)*

AES



ADVANCED ENCRYPTION STANDARD (AES)

- Data Encryption Standard **DES**.
- **Competencia** organizada por el NIST.
- **15 propuestas** presentadas al NIST.
- **Algoritmo de Rijndale (Deaman-Rijmen).**

ADVANCED ENCRYPTION STANDARD RIJNDALE

1

Estructura basada
en
bloques (128 bits)

2

Llave simétrica y
flexible



LLave: Conjunto secreto de números que se usa para cifrar y descifrar un mensaje.

3

Aplicable tanto en
software como en
hardware

4

Seguridad
comprobada

DUALIDAD CRIPTOGRÁFICA DEL AES

AES funciona con dos etapas principales:

Encriptación

Plain text a Cipher text



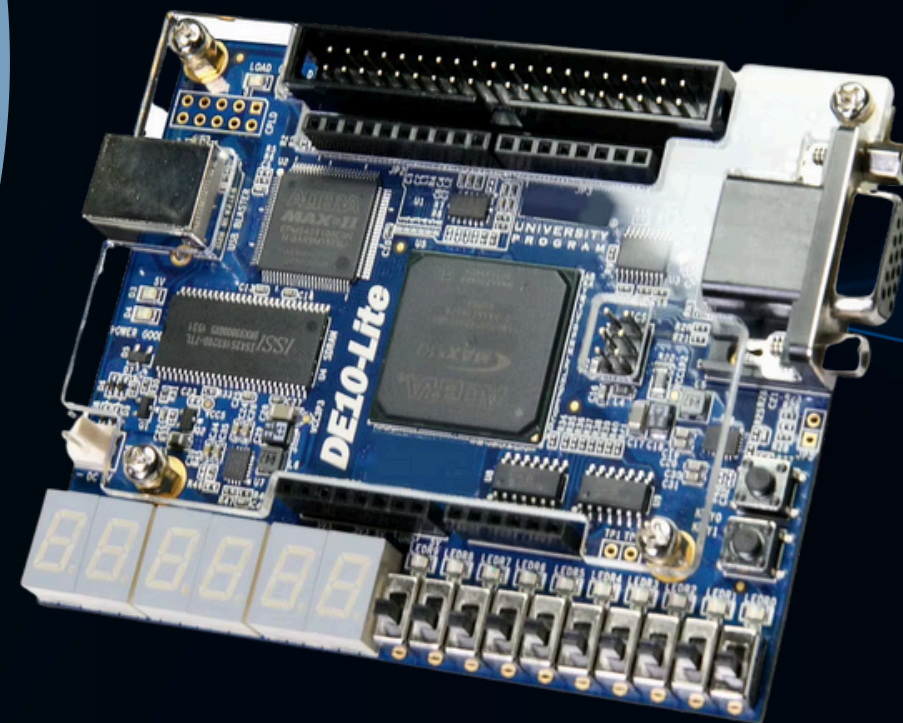
Desencriptación

Cipher text a Plain text



FIELD PROGRAMMABLE GATE ARRAY (FPGA)

Es un chip que no corre software, sino que modifica su **hardware** para realizar tareas específicas.

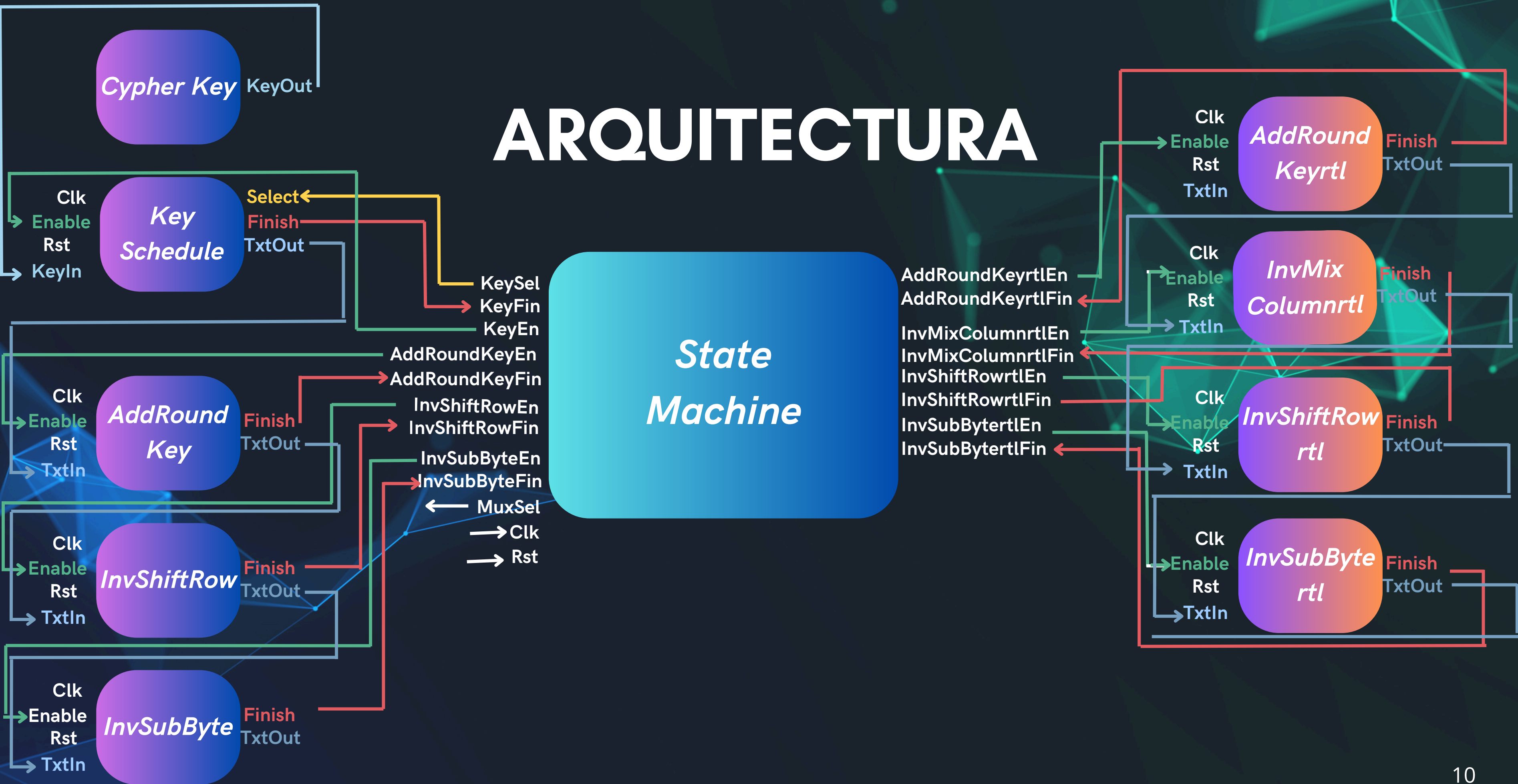


MÓDULOS Y SUBGREGACIÓN

División entre partes del proyecto
y módulos creados



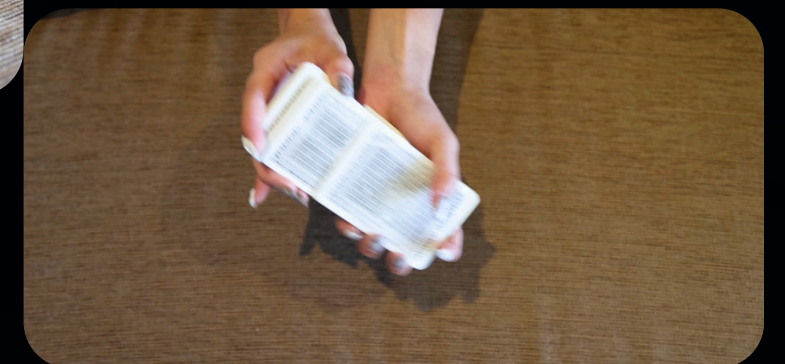
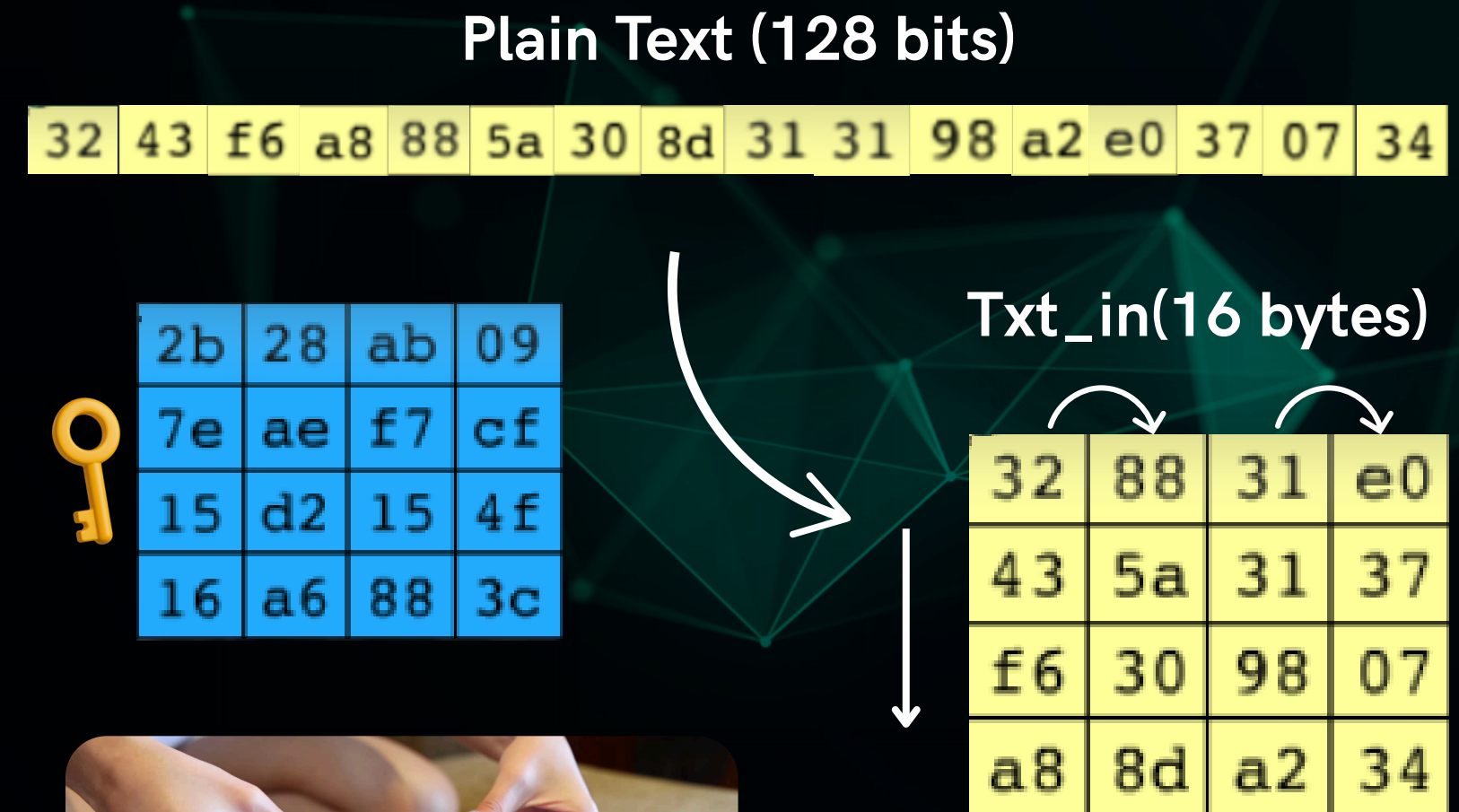
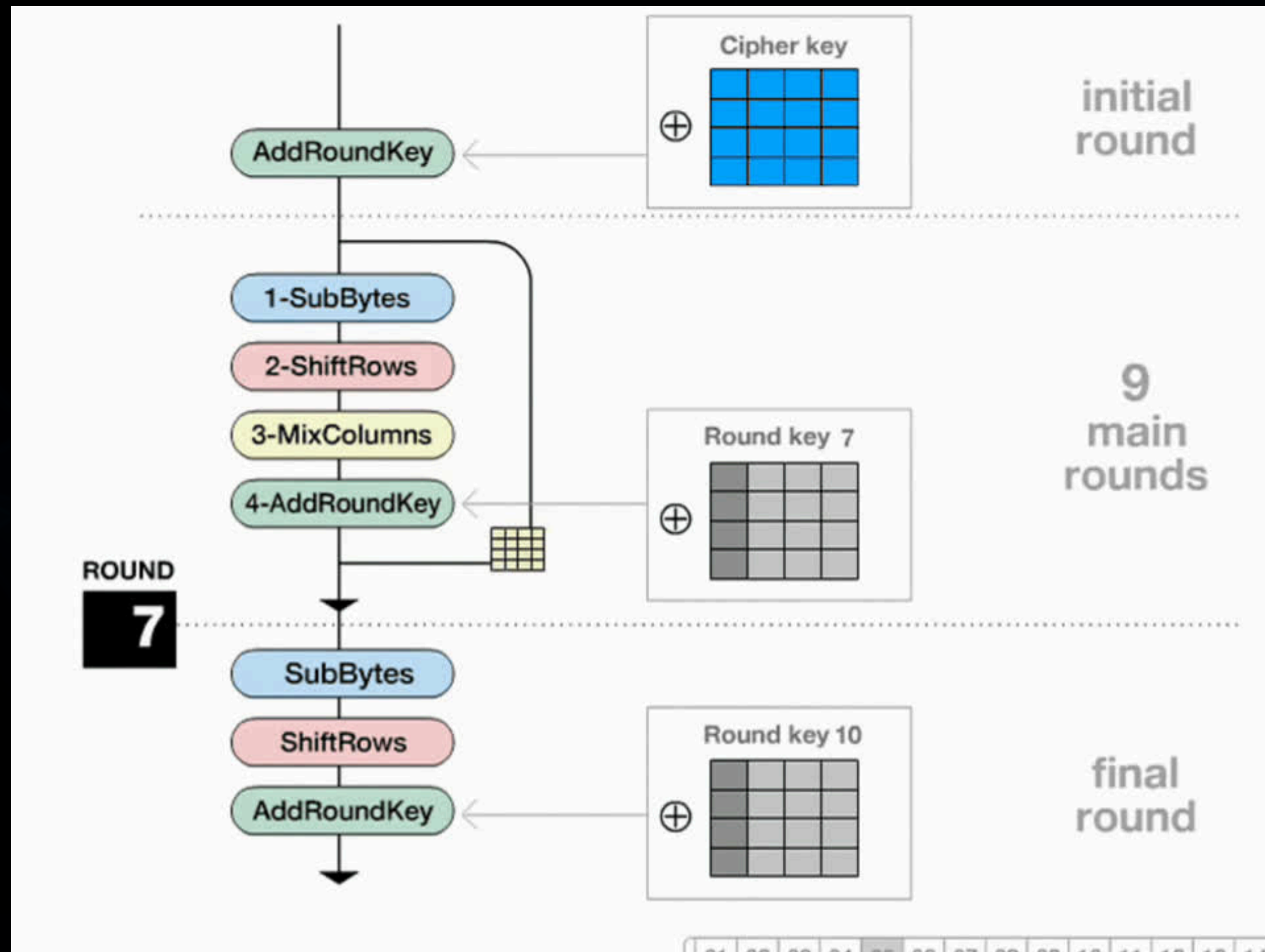
ARQUITECTURA



ENCRYPTACIÓN



PROCESO DE ENCRYPTACIÓN



Fernando Hernandez



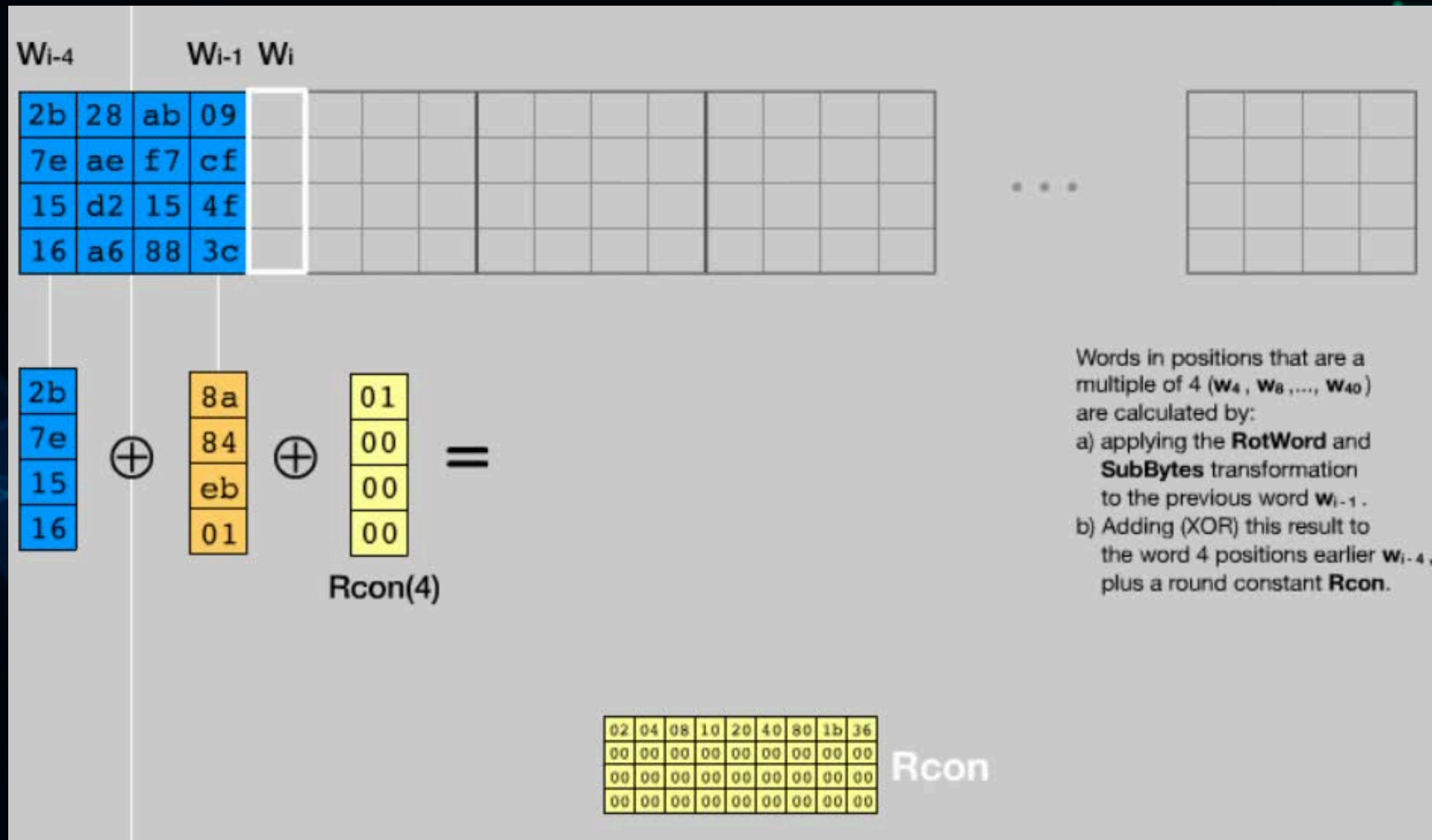
Anell Moreno



Key Schedule



KEY SCHEDULE



Genera las **subclaves** a partir de la clave principal (128 bits). Estas subclaves se usan en cada ronda del AES (con AddRoundKey).

Para cada ronda 1 a 10:
SubBytes + ShiftRows + XOR
 con *Rcon* y palabras anteriores para generar nuevas claves.



Resultado: una subclave nueva por ronda, ¡lista para cifrar!

KEY SCHEDULE

 *Llave inicial
(128 bits)*

*Dividir en 4 palabras (32 bits cada una):
 W_0, W_1, W_2, W_3*

Para $i = 4$ hasta 43:

$W[i] = W[i - 4] \circ W[i - 1] \circ \text{SubWord}$
 $(\text{RotWord}(W[i - 1] \circ \text{Rcon}[i4])$

$i \bmod 4 = 0$

Si

$W[i] = W[i - 4] \circ W[i - 1]$

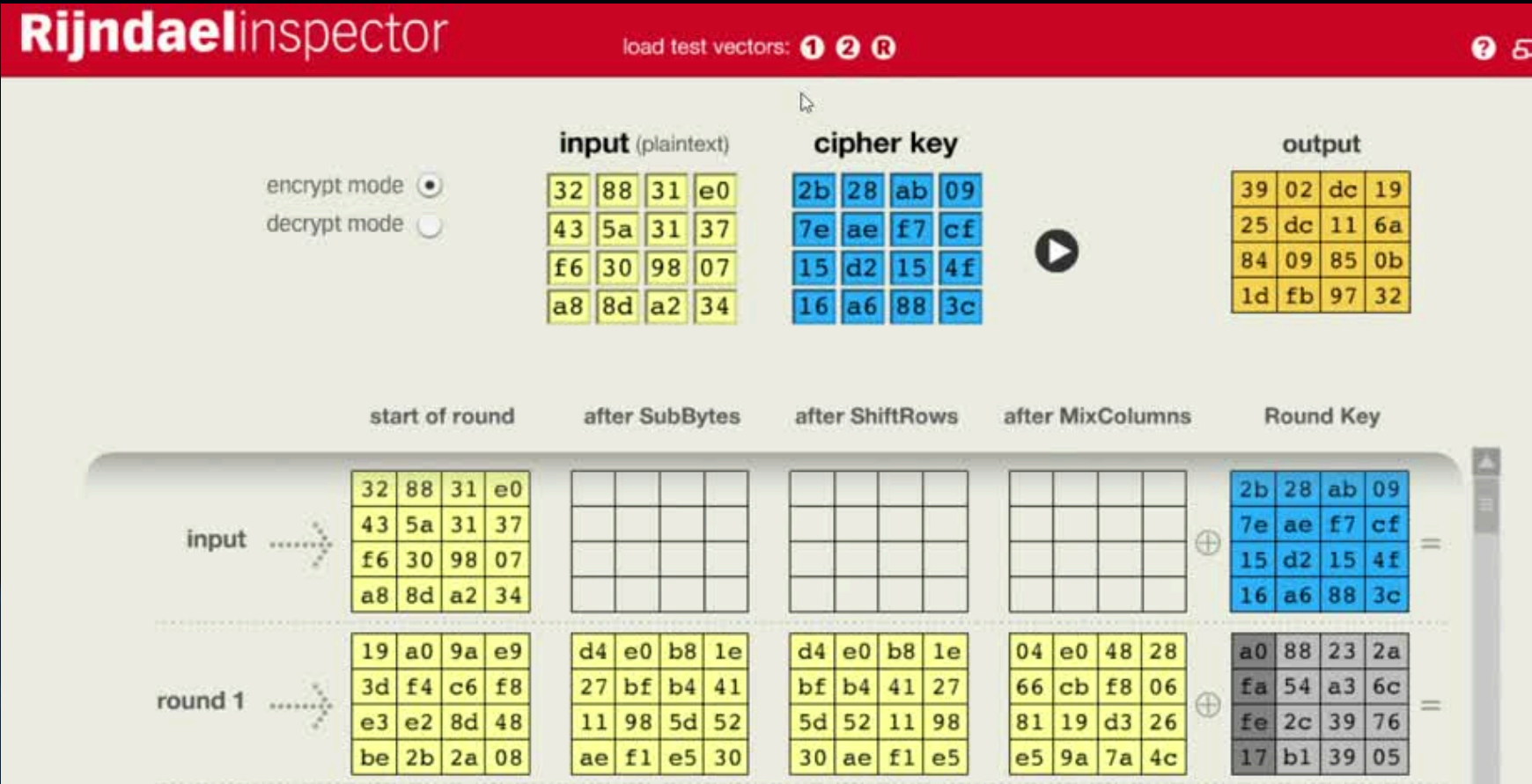
No

*Agrupar cada 4 palabras $\rightarrow$ Round Key
Round Key 0: $W_0 \parallel W_1 \parallel W_2 \parallel W_3$
Round Key 1: $W_4 \parallel W_5 \parallel W_6 \parallel W_7$
...
Round Key 10: $W_{40} \parallel W_{41} \parallel W_{42} \parallel W_{43}$*

Finish

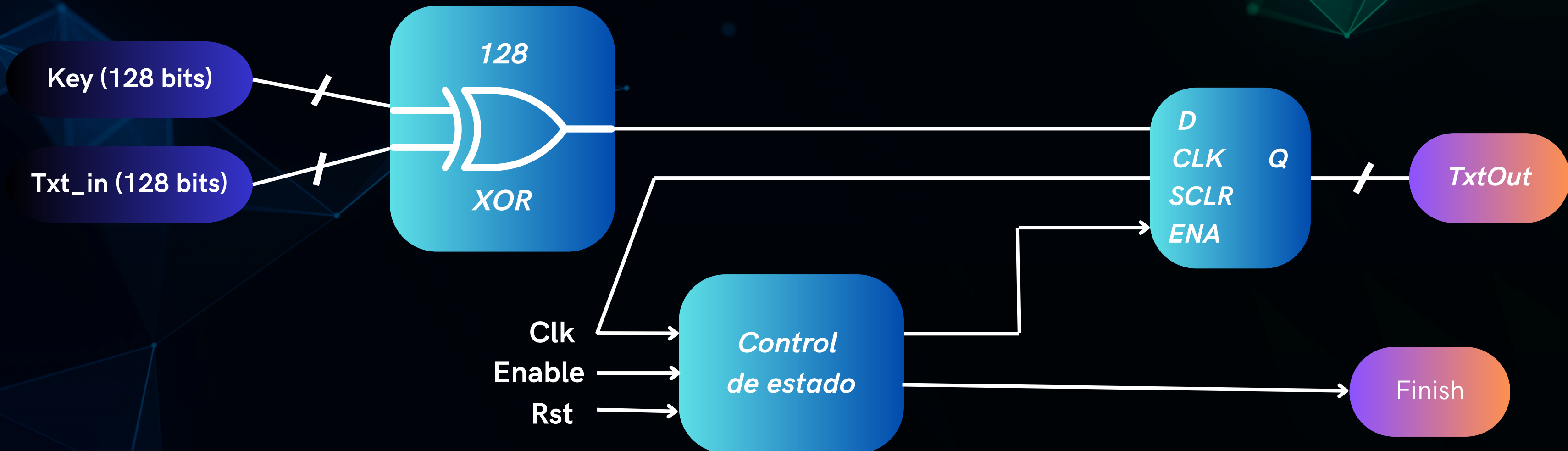
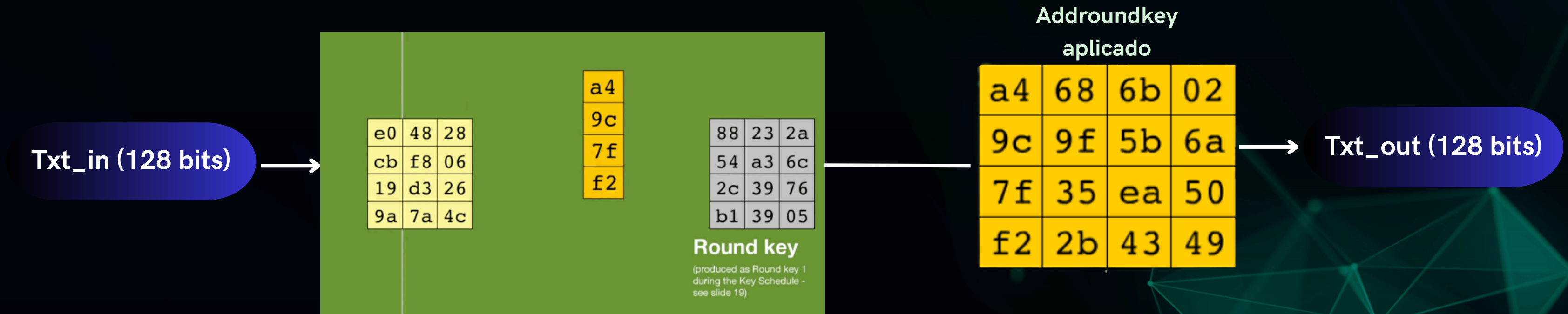
SIMULACIÓN

Key Schedule



/keyschedule_tb/Clk	0						
/keyschedule_tb/Enable	0						
/keyschedule_tb/Rst	0						
/keyschedule_tb/KeyIn	128'h2B7E15162...	2B7E151628AED2A6ABF7158809CF4F3C					
/keyschedule_tb/Sel	4'hA	1		2			
/keyschedule_tb/Finish	1						
/keyschedule_tb/KeyOut	128'hD014F9A8C...	A0FAFE1788542CB123A339392A6C7605	F2C295F27A96B9435935807A7359F67F				

ADDROUNDKEY



SIMULACIÓN

Addroundkey

RijndaelInspector load test vectors: 1 2 R 5

encrypt mode ☒
decrypt mode ☐

input (plaintext)

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

cipher key

2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

output

39	02	dc	19
25	dc	11	6a
84	09	85	0b
1d	fb	97	32

start of round **after SubBytes** **after ShiftRows** **after MixColumns** **Round Key**

input →

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

round 1 →

19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08



2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34



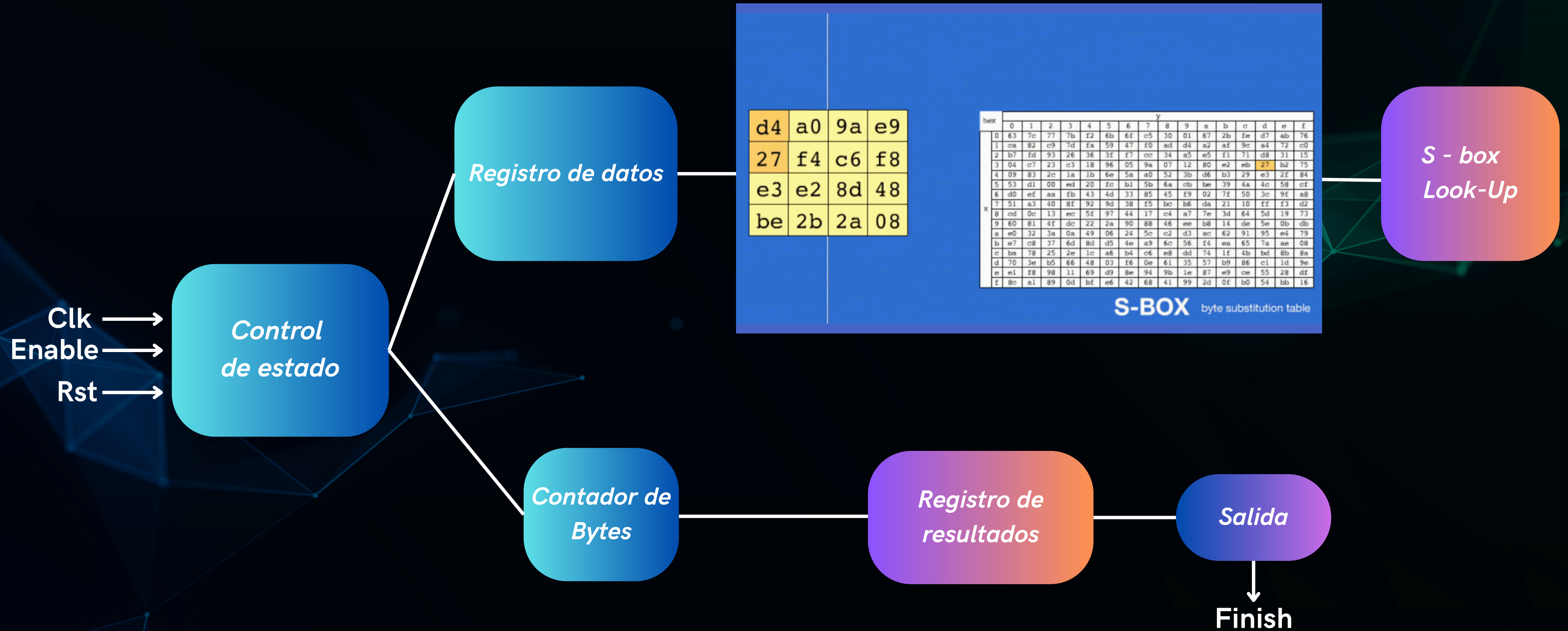
19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

Msgs

/tb_addroundkey/T...	-No Data-
/tb_addroundkey/K...	-No Data-
/tb_addroundkey/T...	-No Data-
/tb_addroundkey/E...	-No Data-
/tb_addroundkey/Clk	-No Data-
/tb_addroundkey/Rst	-No Data-
/tb_addroundkey/Fi...	-No Data-

000000000000000000000000...	3243F6A8885A308D813198A2E0370734
000000000000000000000000...	2B7E151628AED2A6ABF7158809CF4F3C
XXXXXXXXXXXXXXXXXXXXXXXXXXXX	193DE3BEA0F4E22B9AC68D2AE9F84808

SUBBYTES



SIMULACIÓN

Subbytes

RijndaelInspector load test vectors: 1 2 R ? 5

encrypt mode ☒
decrypt mode ☐

input (plaintext)

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

cipher key

2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

output

39	02	dc	19
25	dc	11	6a
84	09	85	0b
1d	fb	97	32

start of round

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

after SubBytes

after ShiftRows

after MixColumns

Round Key

2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

round 1

19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

after SubBytes

d4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30

after ShiftRows

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

after MixColumns

04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

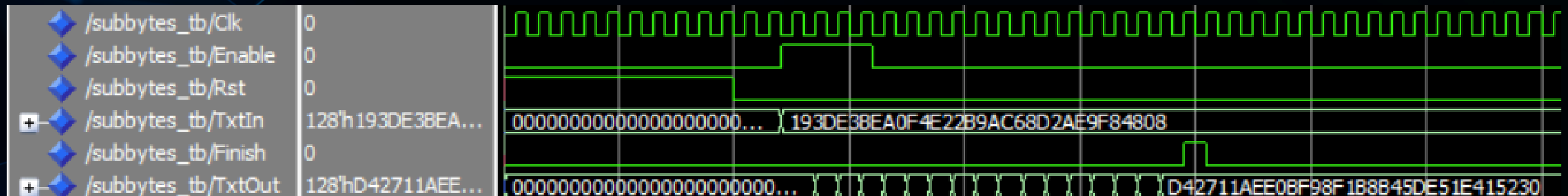
Round Key

a0	88	23	2a
fa	54	a3	6c
fe	2c	39	76
17	b1	39	05

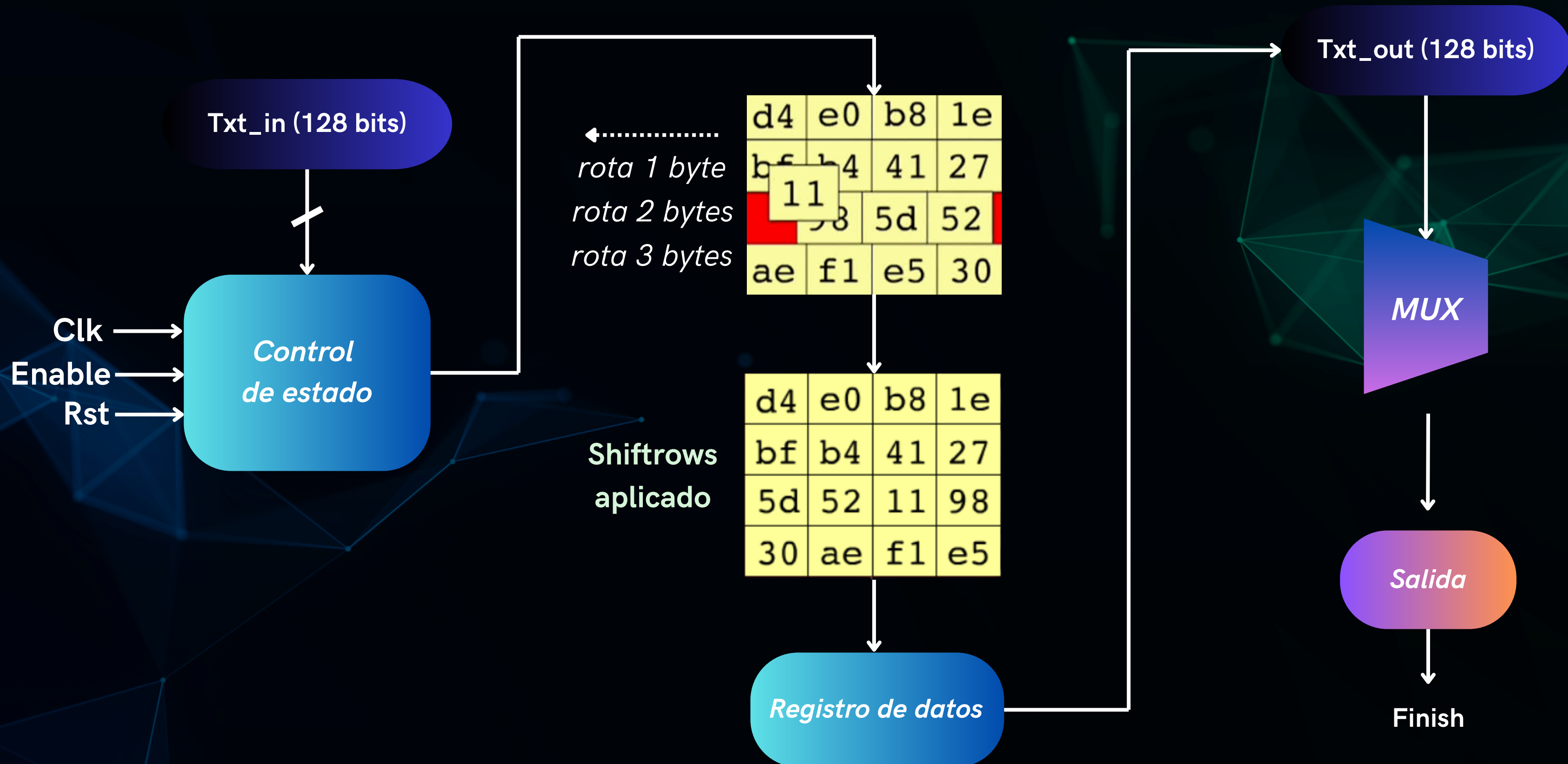
19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

→

d4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30



SHIFTRROWS



SIMULACIÓN

Shiftrows

RijndaelInspector load test vectors: 1 2 3 test vector 1

encrypt mode ☒
decrypt mode ☐

input (plaintext)

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

cipher key

2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

output

start of round after SubBytes after ShiftRows after MixColumns Round Key

input →

round 1 →

d4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

/shiftrows_tb/Clk	1	
/shiftrows_tb/Enable	0	
/shiftrows_tb/Rst	0	
/shiftrows_tb/TxtIn	128'hD42711AEE...	00000000000000000000000000000000 D42711AEE0BF98F1B8B45DE51E415230
/shiftrows_tb/Finish	0	
/shiftrows_tb/TxtOut	128'hD4BF5D30E...	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX D4BF5D30E0B452AEB84111F11E2798E5

MIX COLUMNS

SUMA

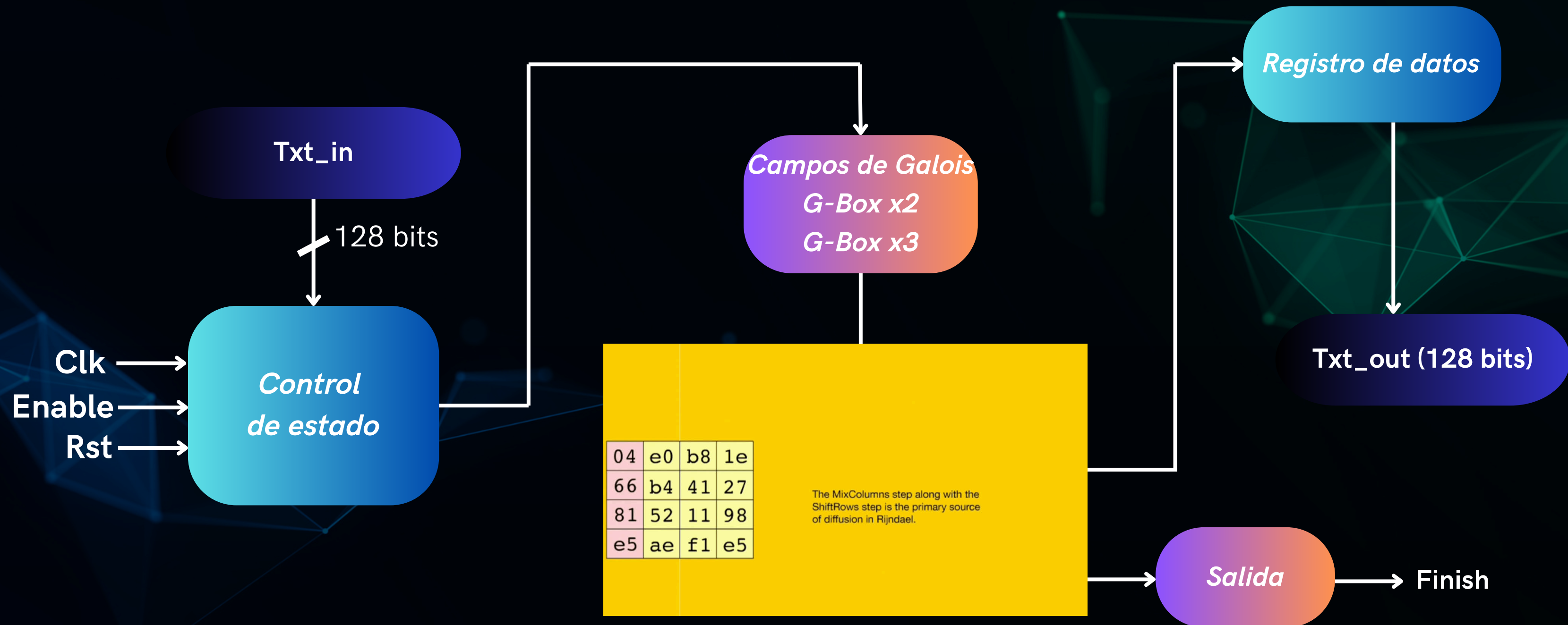
Se hacen con XOR (suma binaria sin acarreo).

MULTIPLICACIÓN

Polinomio en Binario



MIX COLUMNS



SIMULACIÓN

Mix Column

RijndaelInspector load test vectors: 1 2 R

encrypt mode ☒
decrypt mode ☐

input (plaintext)

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

cipher key

2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

output

39	02	dc	19
25	dc	11	6a
84	09	85	0b
1d	fb	97	32

start of round **after SubBytes** **after ShiftRows** **after MixColumns** **Round Key**

input

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

round 1

19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

d4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

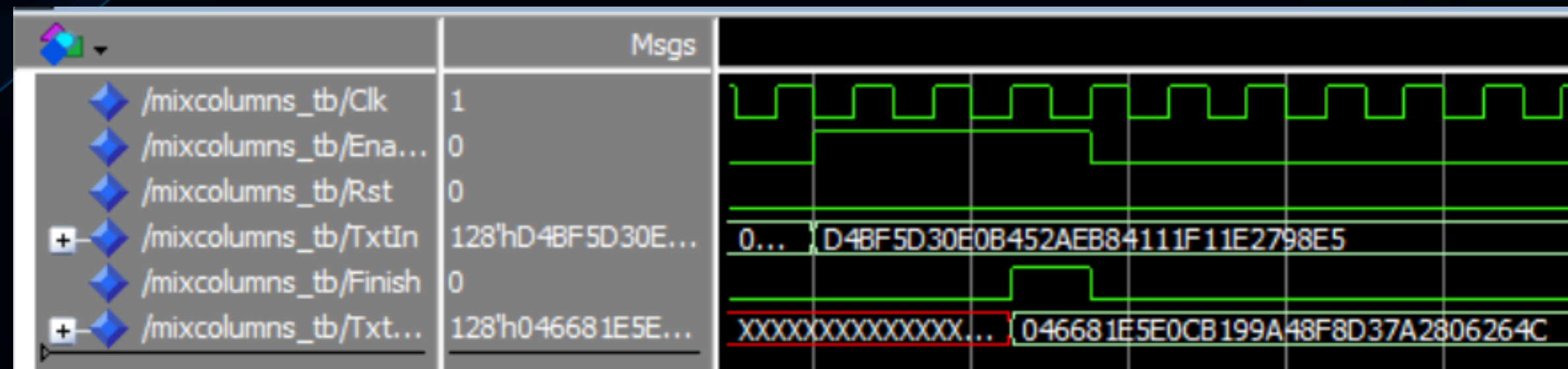
04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

a0	88	23	2a
fa	54	a3	6c
fe	2c	39	76
17	b1	39	05

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

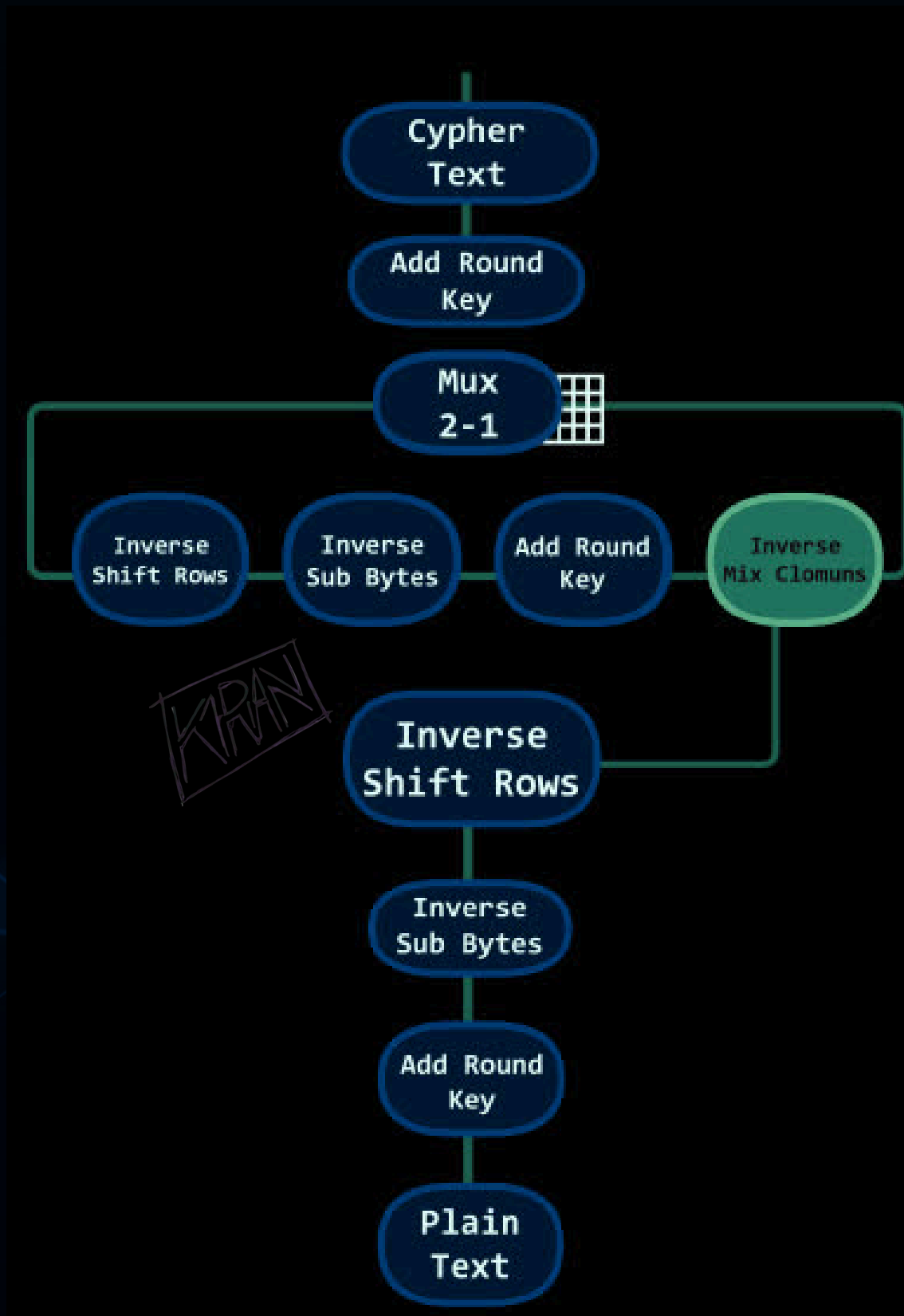
04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c



DESENCRIPTACIÓN

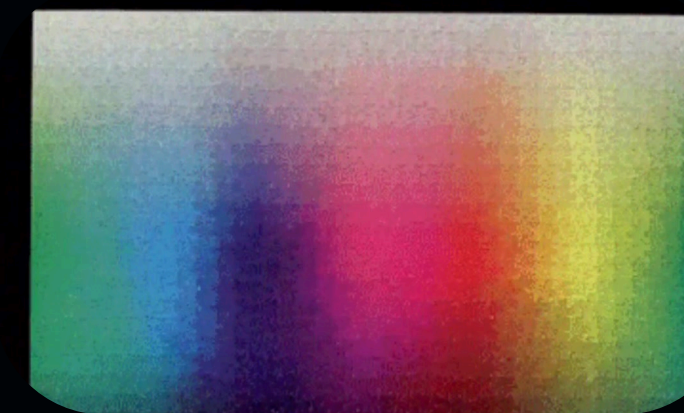


PROCESO DE DESENCRIPTACIÓN



- El texto cifrado se desarma paso a paso hasta recuperar el original.
- Se usan las mismas claves, pero desde la última hasta la primera.
- Se aplican las operaciones contrarias al cifrado:

InvSubBytes → *InvShiftRows* → *InvMixColumns* → *InvAddRoundKey*.





Líder

Omar Reyes
Barrueta



Líder

Arturo Balboa
Alvarado

Inv SubBytes

Matías Piedra Pichardo
Josemaría De Alba
Arguelles
Ricardo Ruiz Cano
Samantha Erin Medina
Muñoz
Fernando García Padilla

Inv ShiftRows

Socorro Alejandra Montelongo
Loredo
José Fernando Vivanco
Benavides
Juan César Nuño Rivera
Gabriel Enrique Lascurain
Flores
Arturo Uriás Jiménez

Chat
General

Inv AddRoundKey

Arturo Balboa Alvarado
Anell Moreno Quilantán
Omar Reyes Barrueta
Mario Armando Moscoso
Becerra
Fernando Hernández
Hernández

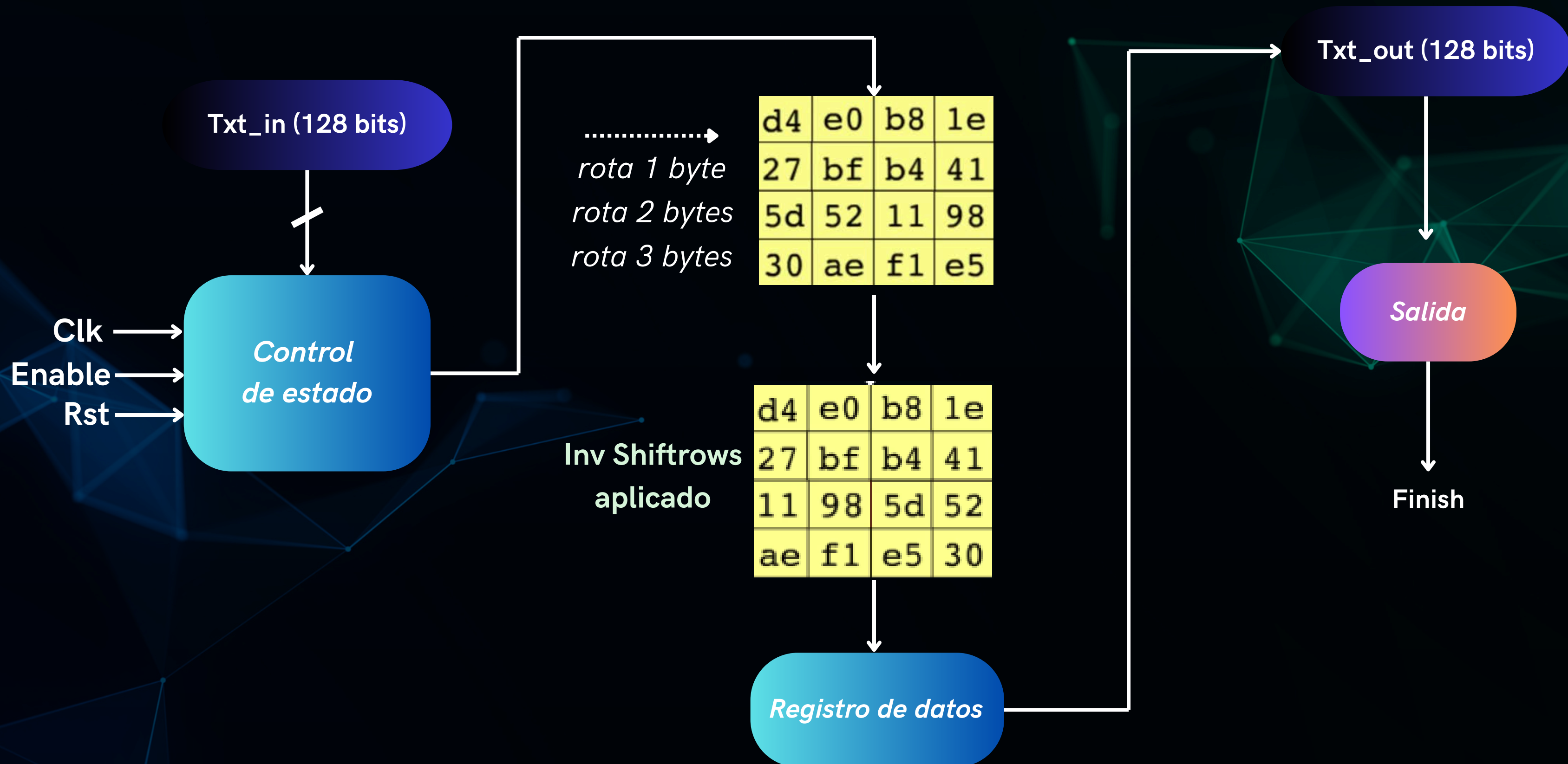
Inv MixColumns

Arturo Balboa
Alvarado
Omar Ryes
Barrueta

Key Schedule

Anell Moreno Quilantán
Mario Armando Moscoso
Becerra
Fernando Hernández
Hernández

INV SHIFTRROWS



SIMULACIÓN

Inv Shiftrows

RijndaelInspector load test vectors: 1 2 R 5

encrypt mode ☒
decrypt mode ☐

input (plaintext)

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

cipher key

2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

output

39	02	dc	19
25	dc	11	6a
84	09	85	0b
1d	fb	97	32

start of round

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

after SubBytes

after ShiftRows

after MixColumns

Round Key

2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

input

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

round 1

19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

after SubBytes

d4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30

after ShiftRows

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

after MixColumns

04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

Round Key

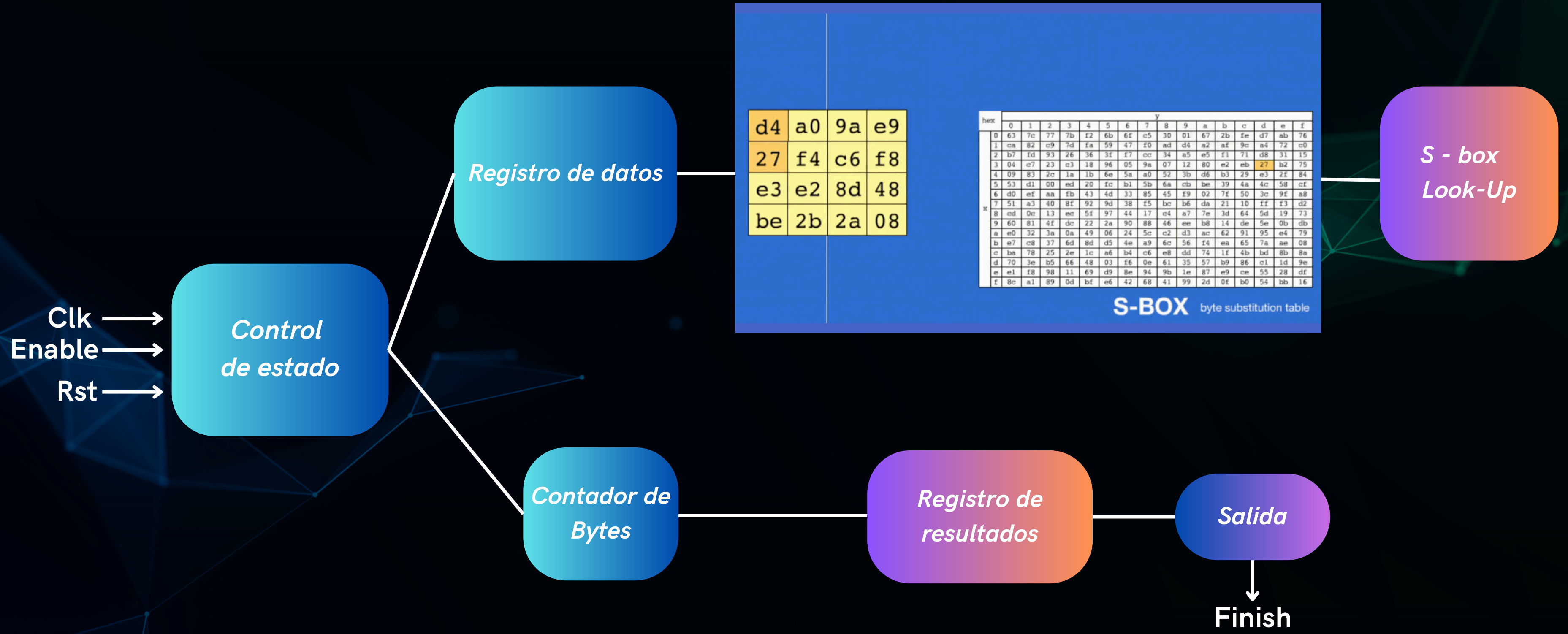
a0	88	23	2a
fa	54	a3	6c
fe	2c	39	76
17	b1	39	05

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

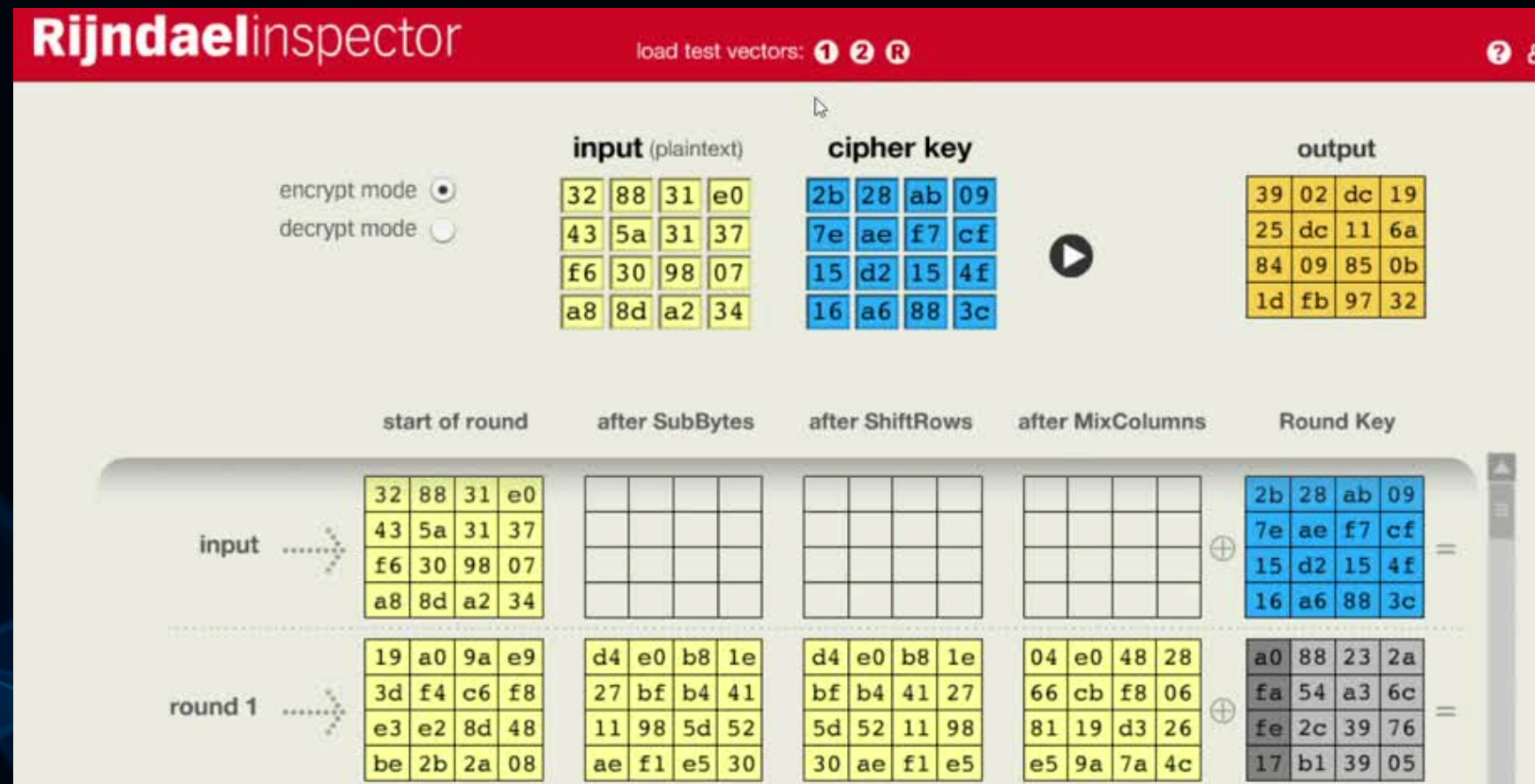
d4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30

	Msgs	
/inverse_shift_rows...	0	
/inverse_shift_rows...	0	
/inverse_shift_rows...	0	
+ /inverse_shift_rows...	128'hD48F5D30E...	D48F5D30E0B452AEB8411F11E2798E5
/inverse_shift_rows...	0	
+ /inverse_shift_rows...	128'hD42711AEE...	D42711AEE0BF98F1B8B45DE51E415230

INV SUBBYTES

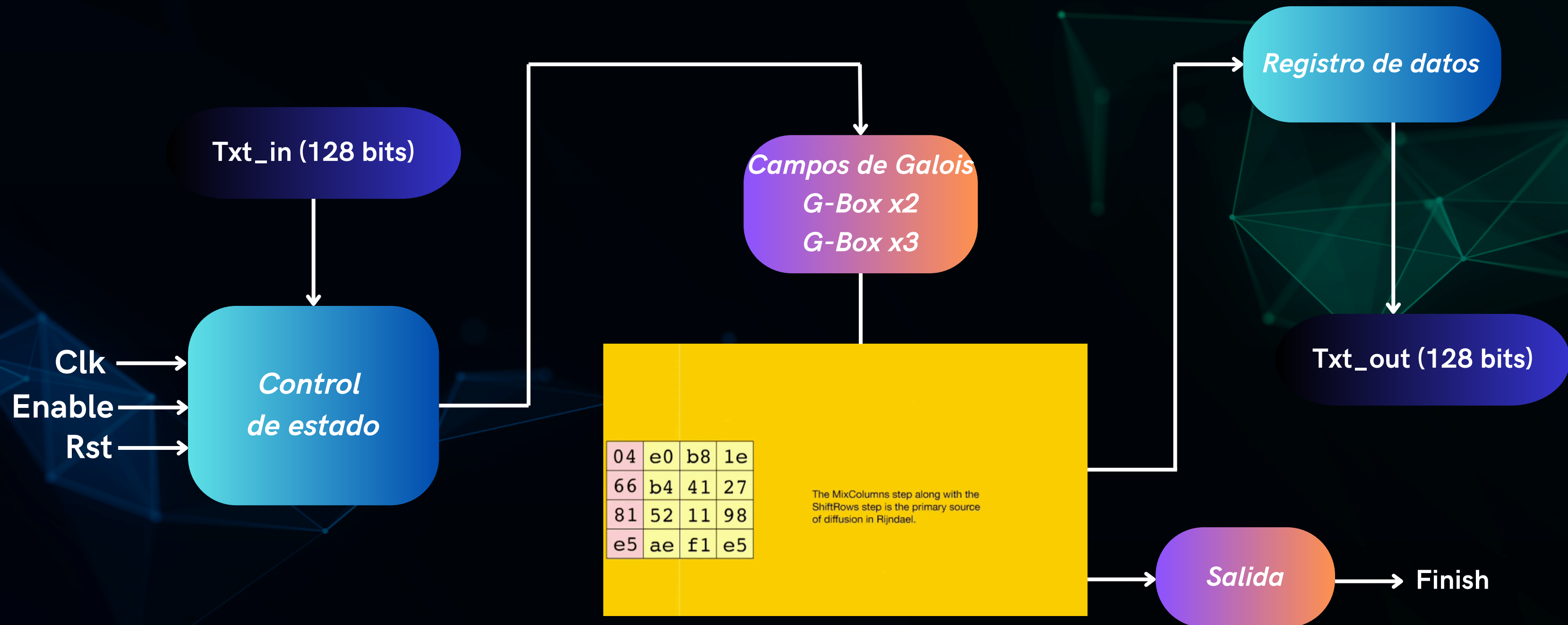


Inv Subbytes



	/invsubbytes_tb/Clk	1				
	/invsubbytes_tb/En...	1				
	/invsubbytes_tb/Rst	0				
	 /invsubbytes_tb/TxtIn	128'h000000000...	D42711AEE0BF98F1B8B45DE51E415230			
	/invsubbytes_tb/Fin...	1				
	 /invsubbytes_tb/Tx...	128'h525252525...	193DE38EA0F4E22B9AC68D2AE9F84808			

INV MIX COLUMN



SIMULACIÓN

Inv Mix Column

RijndaelInspector load test vectors: 1 2 R ? 5

encrypt mode ☒
decrypt mode ☐

input (plaintext)

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

cipher key

2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

output

39	02	dc	19
25	dc	11	6a
84	09	85	0b
1d	fb	97	32

start of round **after SubBytes** **after ShiftRows** **after MixColumns** **Round Key**

input

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

round 1

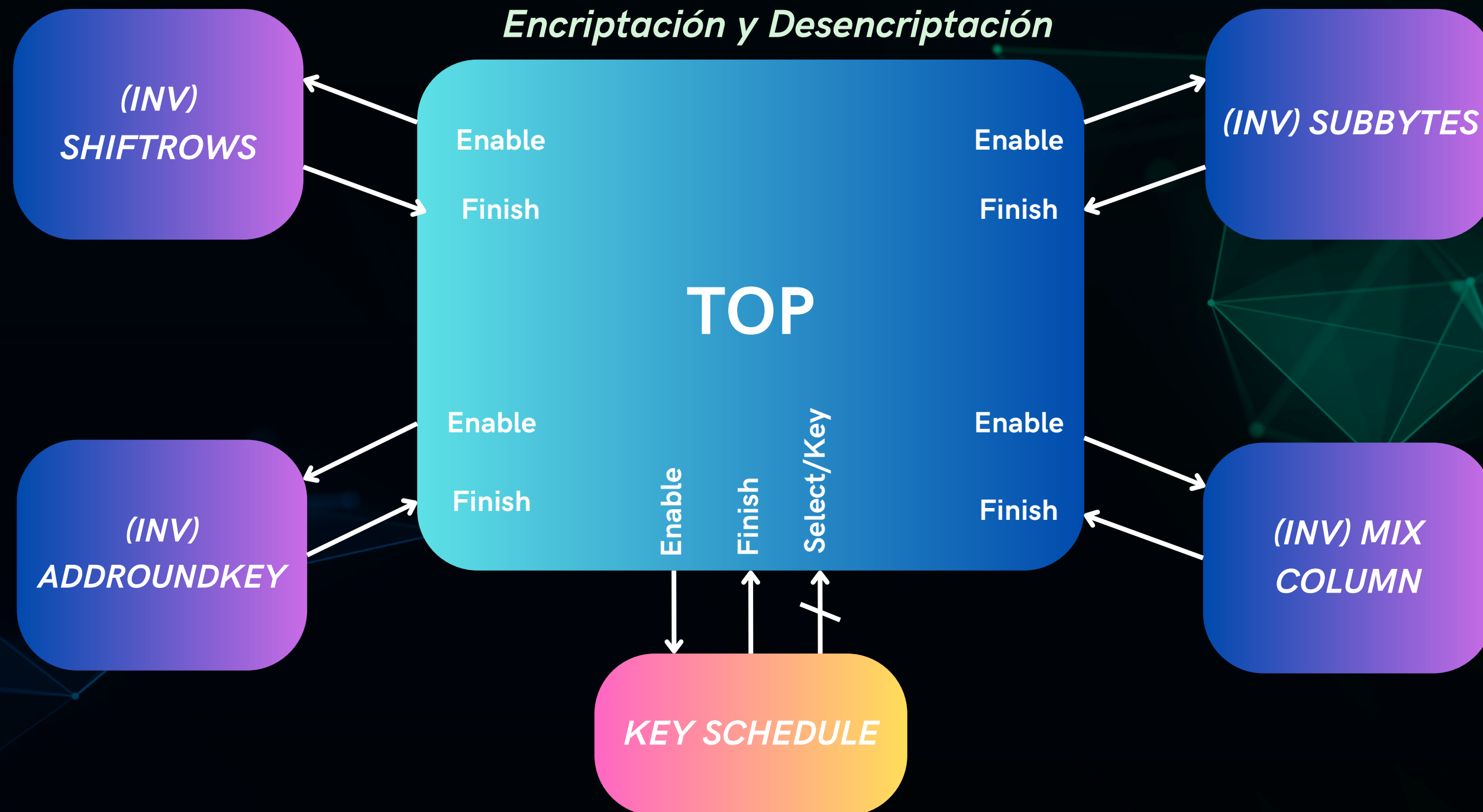
19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

046681E5E0CB199A48F8D37A2806264C	00000000
00000...	D4BF5D30E0B452AEB8411F11E2798E5

MÁQUINA DE ESTADOS



PROCESO ENCRIPTADO:

PRIMERA RONDA: ADD ROUND KEY

- 9 RONDAS: SUBBYTES, SHIFTRROWS, MIXCOLUMNS, ADDROUNDKEY
- RONDA FINAL: SUBBYTES, SHIFTRROWS, ADDROUNDKEY

PROCESO DESENCRIPTADO:

PRIMERA RONDA: ADD ROUND KEY

- 9 RONDAS: INVSHIFTRROWS, INVSUBBYTES, INVADDROUNDKEY, INVMIXCOLUMNS, 35
- RONDA FINAL: INVSHIFTRROWS, INV SUBBYTES, , INVADDROUNDKEY

PRUEBA FINAL

input (plaintext)

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

cipher key

2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

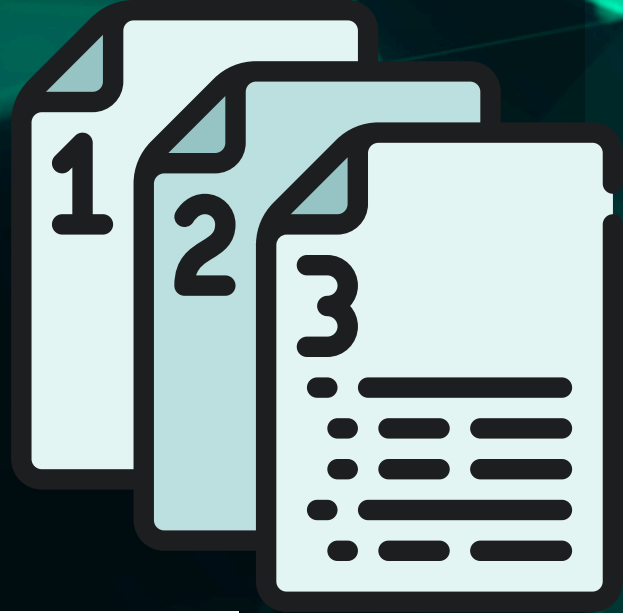
output

Encriptación

/top_tb/CipherKeyI...	128'h2B7E15162...	2B7E151628AED2A6ABF7158809CF4F3C
/top_tb/PlainText	128'h3243F6A88...	3243F6A8885A308D313198A2E0370734
/top_tb/CypherText	128'h3925841D0...	3925841D02DC09FBDC118597196A0B32

Desencriptación

/top_tb/CipherKeyI...	128'h2B7E15162...	2B7E151628AED2A6ABF7158809CF4F3C
/top_tb/CypherText	128'h3925841D0...	3925841D02DC09FBDC118597196A0B32
/top_tb/PlainText	128'h3243F6A88...	3243F6A8885A308D313198A2E0370734



CONTROL DE VERSIONES

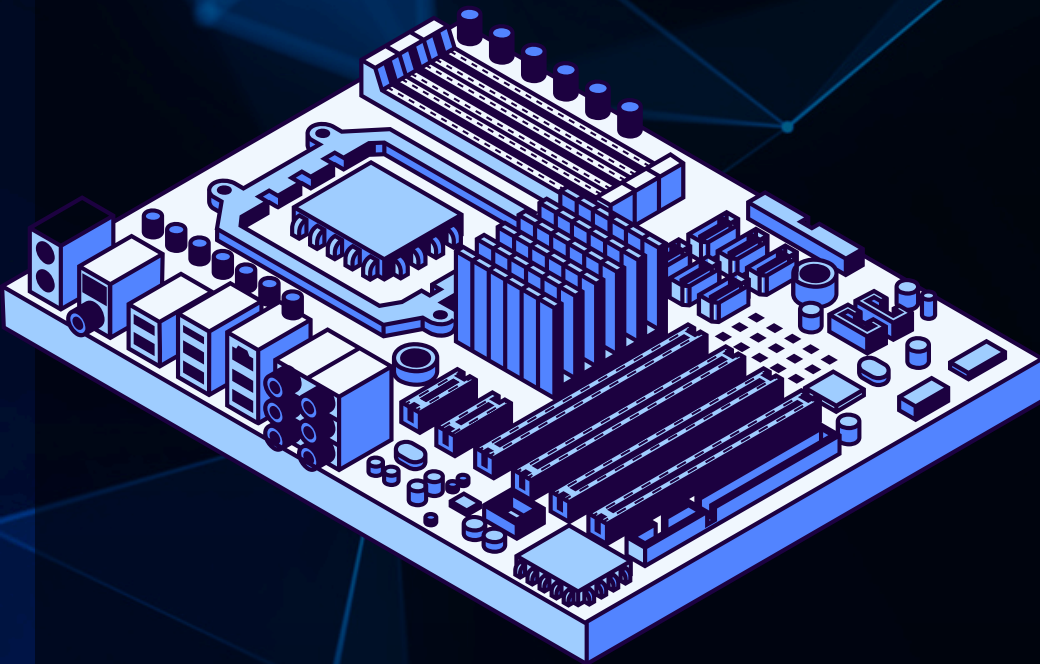
<https://github.com/Matziaz/TE2002B>



Scan me!

BENCHMARK

Software VS Hardware



SOFTWARE



BENCHMARK

Python

```
118
119 def aes_encrypt(data, key):
120     expanded_key = key_expansion(key)
121     blocks = [data[i:i+16] for i in range(0, len(data), 16)]
122     encrypted_blocks = [aes_encrypt_block(block, expanded_key) for block in blocks]
123     return b''.join(encrypted_blocks)
124
125 # ----- Entry Point -----
126
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\omar\OneDrive\Escritorio\Cpp y Python implementations (Benchmarking)\Python> & C:/Users/omar/AppData/Local/Programs/Python/AES_Python.py
Enter a string to encrypt: 328831e0435a3137f6309807a88da234
Encrypted result (hex): 0652d747c574f46b2cabb90046a849256f5664cad4602f32394e3f575b2868925f95ca810f7e9f44cfca37cfcb77d8f2
Tiempo de ejecución: 10.9226 segundos
PS C:\Users\omar\OneDrive\Escritorio\Cpp y Python implementations (Benchmarking)\Python> 
```

Tiempo de ejecución: 10.9226 segundos

BENCHMARK

C++

```
C:\Users\omar\OneDrive\Escr x + v
Enter a string to encrypt: 328831e0435a3137f6309807a88da234

Plaintext in (hex):
33 32 38 38 33 31 65 30 34 33 35 61 33 31 33 37
66 36 33 30 39 38 30 37 61 38 38 64 61 32 33 34

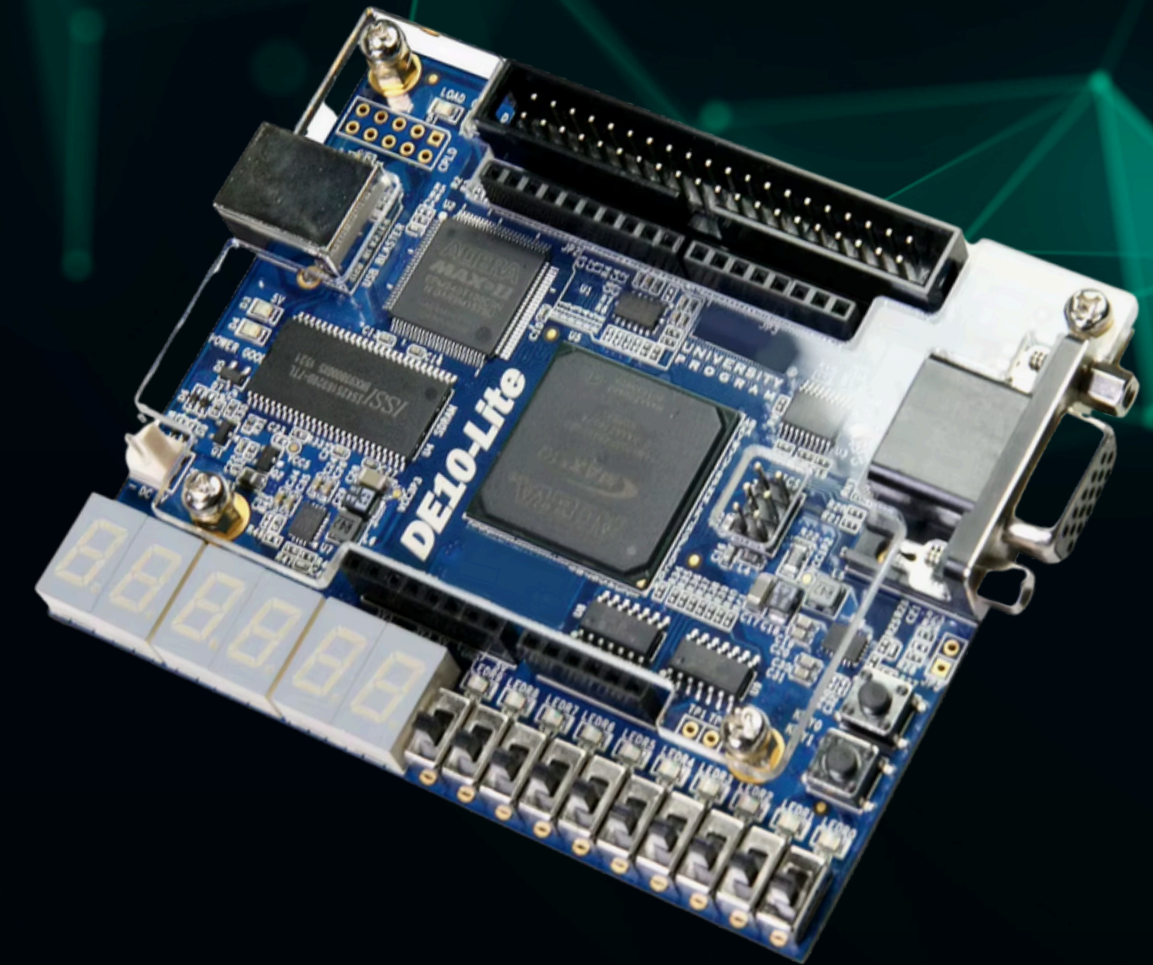
Encrypted result (hex):
6a 06 5d 30 13 4d ca f9 cb 6e 55 3a 69 2f ee b0
3f e3 d3 f1 c3 a9 17 25 6f 66 6f a0 24 e4 c6 8c

Encryption completed in 0 microseconds.

-----
Process exited after 1.729 seconds with return value 0
Presione una tecla para continuar . . . |
```

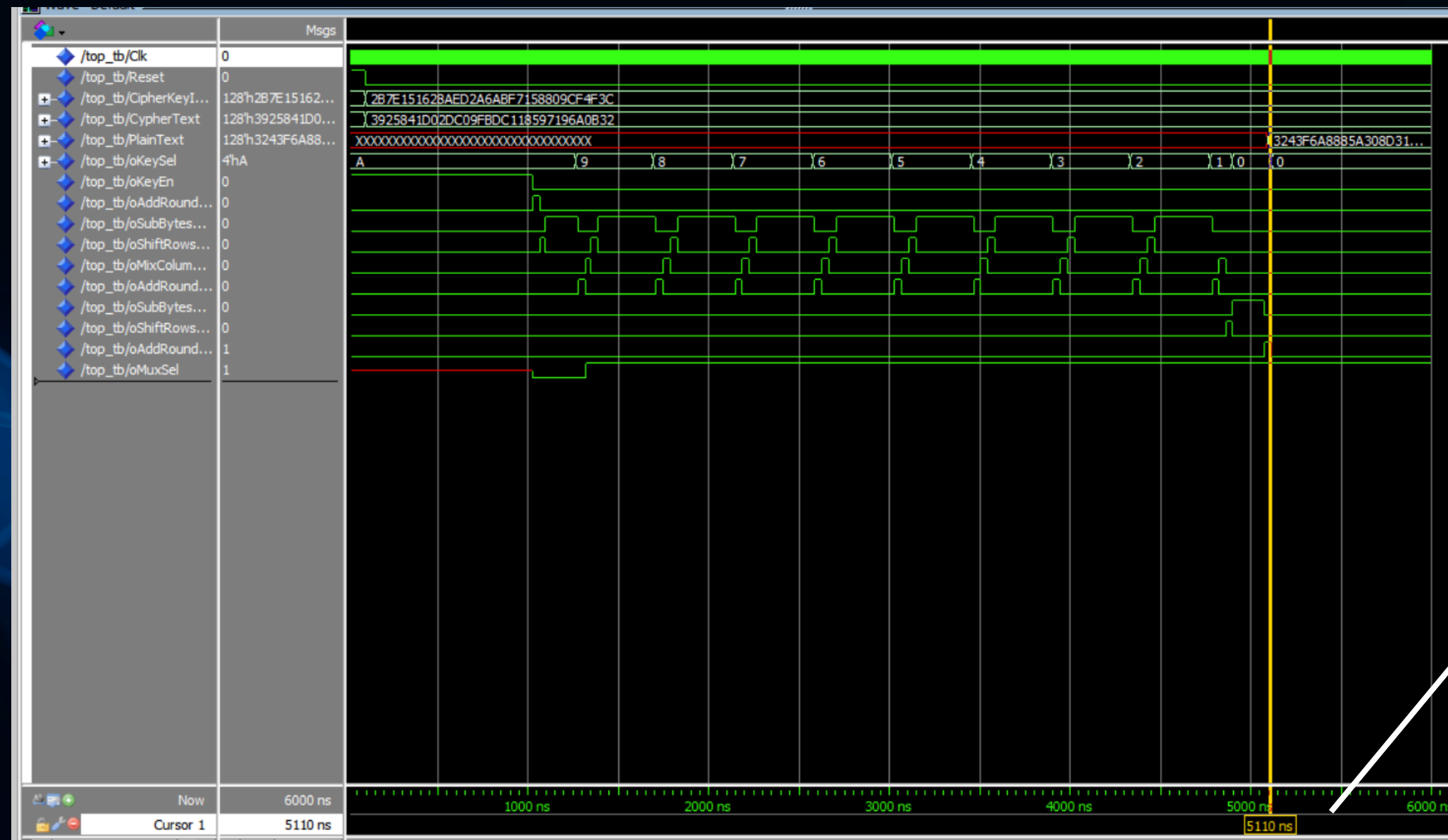
Process exited after 1.729 seconds with return value 0

HARDWARE



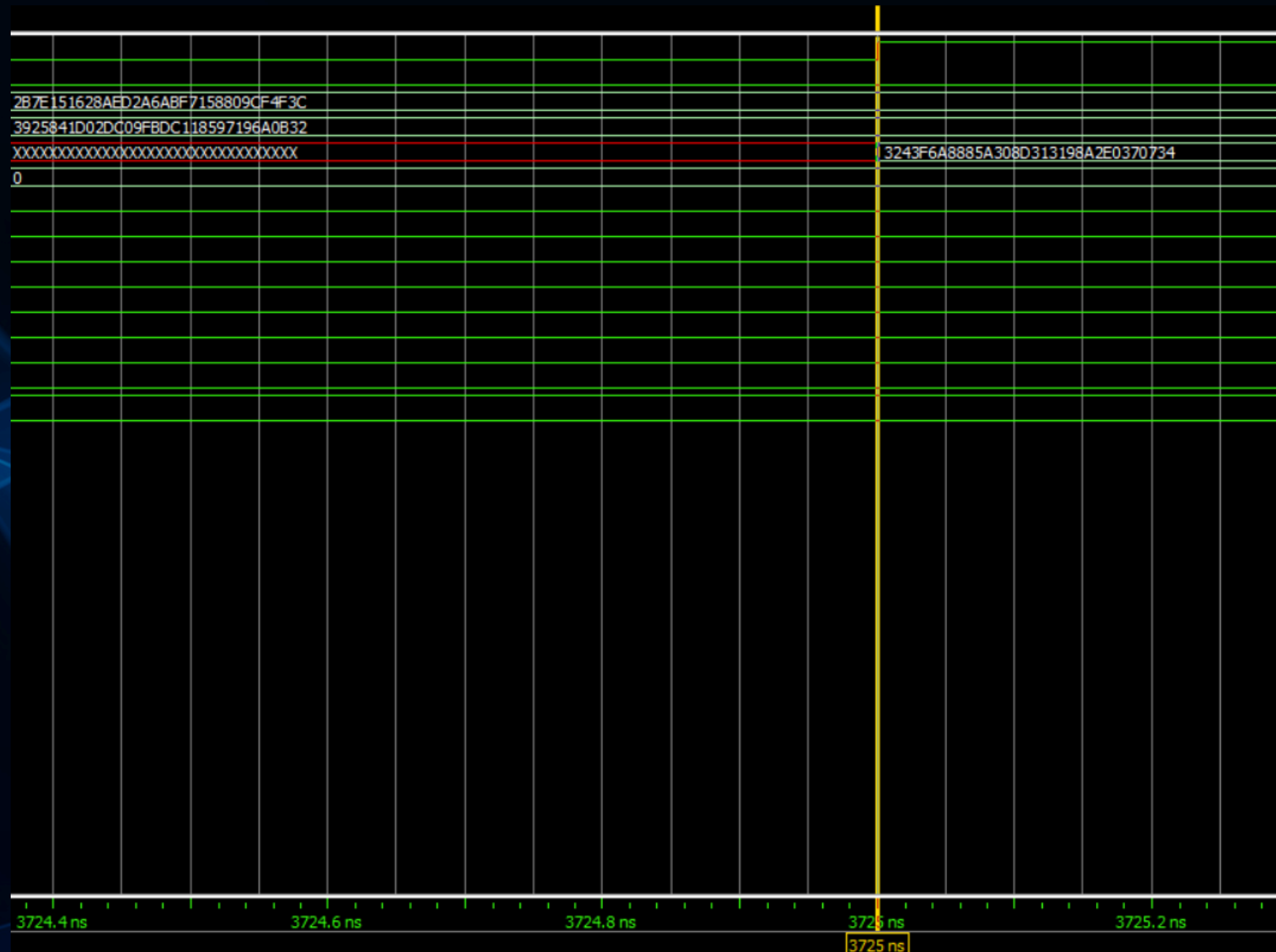
BENCHMARK

VHDL



5110 ns

BENCHMARK



FPGA's
VHDL

Optimizado



Flow Status	Successful - Tue Apr 29 12:49:36 2025
Quartus Prime Version	21.1.0 Build 842 10/21/2021 SJ Lite Edition
Revision Name	AES_Decryption
Top-level Entity Name	Top
Family	MAX 10
Device	10M50DAF672C7G
Timing Models	Final
Total logic elements	5,942 / 49,760 (12 %)
Total registers	3970
Total pins	400 / 500 (80 %)
Total virtual pins	0
Total memory bits	0 / 1,677,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 288 (0 %)
Total PLLs	0 / 4 (0 %)
UFM blocks	0 / 1 (0 %)
ADC blocks	0 / 2 (0 %)

BENCHMARK

FPGA's
VHDL

Optimizado

Timing Models	Final
Total logic elements	5,942 / 49,760 (12 %)

12%!!

¿QUIÉN ES EL MÁS RÁPIDO?

Comparado contra...	Veces más rápido fue VHDL 2 (3725 ns)
Python (10.97 s)	~2,945,638 veces
C++ (1.729 s)	~464,431 veces
VHDL 1 (5110 ns)	~1.372 veces

¡FPGA LO DEMUESTRA! **altera**[™]
An Intel Company

30 MENTES, 1 MISIÓN:

ASEGURAR EL MENSAJE, DE IDA Y DE VUELTA



¡MUCHAS GRACIAS!

BIBLIOGRAFIA

EITCA. (2023) ¿Cuál es el papel del polinomio irreducible en la operación de multiplicación en los campos de Galois?
<https://eitca.org/cybersecurity/eitc-is-ccf-classical-cryptography-fundamentals/aes-block-cipher-cryptosystem/introduction-to-galois-fields-for-the-aes/examination-review-introduction-to-galois-fields-for-the-aes/what-is-the-role-of-the-irreducible-polynomial-in-the-multiplication-operation-in-galois-fields/>

EITCA. (2023). ¿Cómo se realiza la multiplicación en Galois Fields en el contexto del algoritmo AES?
<https://es.eitca.org/cybersecurity/eitc-is-ccf-classical-cryptography-fundamentals/aes-block-cipher-cryptosystem/introduction-to-galois-fields-for-the-aes/examination-review-introduction-to-galois-fields-for-the-aes/how-is-multiplication-performed-in-galois-fields-in-the-context-of-the-aes-algorithm/>