

CASA

CYBER SECURITY IN THE AGE
OF LARGE-SCALE ADVERSARIES

HAWKEYE

Recovering Symmetric Cryptography from Hardware Circuits

Gregor Leander¹, Christof Paar², Julian Speith², Lukas Stennes¹

¹ *Ruhr University Bochum (RUB)*

² *Max Planck Institute for Security and Privacy (MPI-SP)*

RUHR
UNIVERSITÄT
BOCHUM

RUB

Gefördert durch

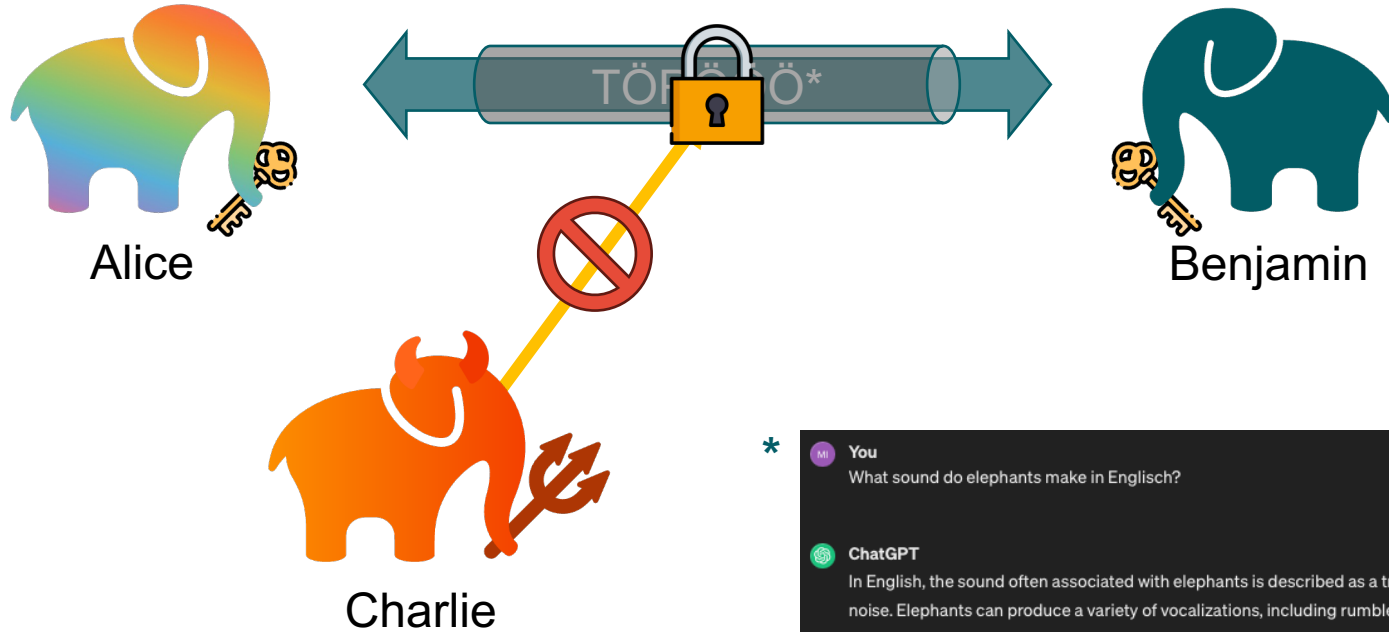
DFG

Deutsche
Forschungsgemeinschaft

MAX PLANCK INSTITUTE
FOR SECURITY AND PRIVACY



SYMMETRIC CRYPTOGRAPHY



*

MI

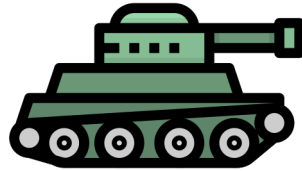
You

What sound do elephants make in English?

ChatGPT

In English, the sound often associated with elephants is described as a trumpet or trumpet-like noise. Elephants can produce a variety of vocalizations, including rumbles, roars, grunts, and trumpets, depending on the context and their emotional state. The trumpet sound is often produced by exhaling through the trunk, and it can vary in pitch and duration.

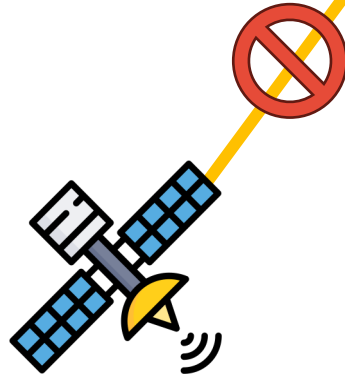
PRACTICAL APPLICATION



Alice



Benjamin

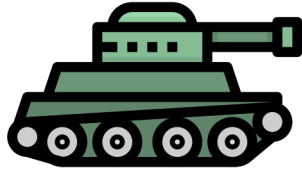


Charlie

What if Charlie still wants to listen in?

- Crypto protocol might be secret
- Charlie may get their hands on a radio
- Algorithm must be recovered for subsequent analysis
- But how?

PRACTICAL APPLICATION



Alice

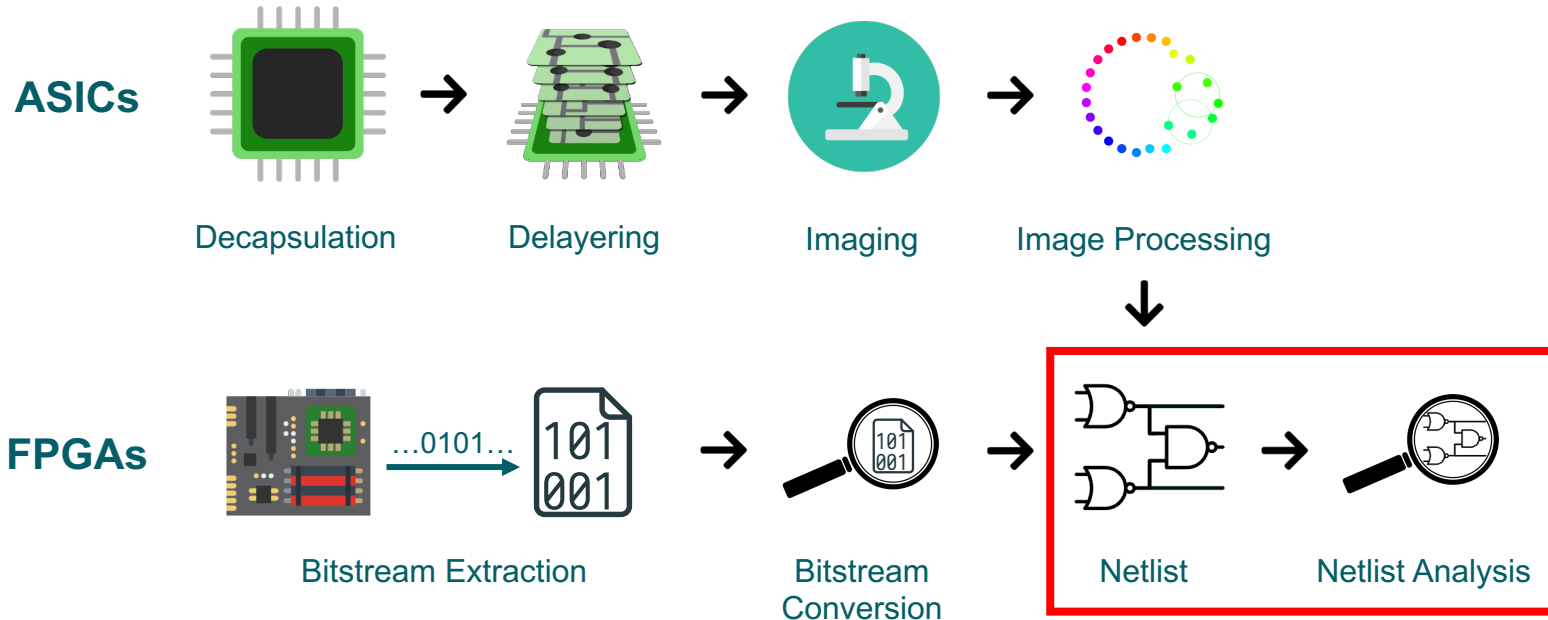


Benjamin

Benjamin still wants to listen in?
Alice's key might be secret
Benjamin can't get their hands on a radio
The key might be recovered for
analysis

- But how?

HRE OVERVIEW



AES

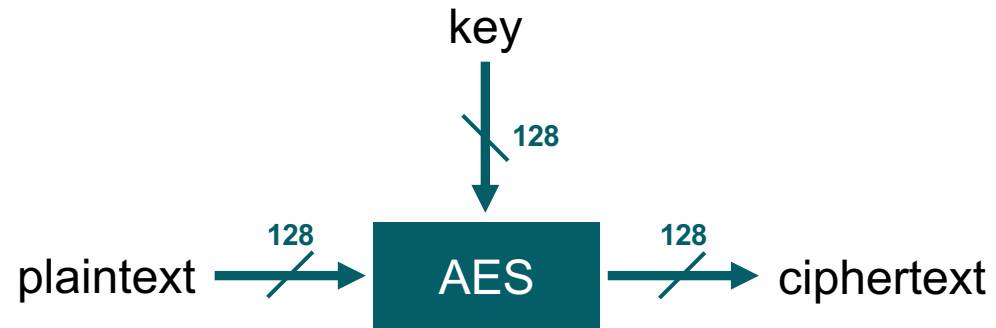
The chalkboard diagram illustrates the components of an AES round function. On the left, a vertical flowchart shows the sequence: **A** (128 bits) → **Byte Sub** → **Shift Row** → **MixCol** → **Key Add**. A bracket on the left labels this entire sequence as "one round".

The central diagram shows the internal data flow. The 128-bit input **A** is divided into four 32-bit lanes: **A₀**, **A₁**, **A₂**, and **A₃**. Each lane is further divided into four 8-bit bytes. The **Byte Sub** stage uses S-boxes (represented by circles with 'S') to substitute each byte. The **Shift Row** stage shifts the bytes in each row. The **MixCol** stage mixes the columns, with a note indicating it uses the "last bytes of subkey". The **Key Add** stage performs XOR operations with subkeys **K_{i,0}** and **K_{i,16}**.

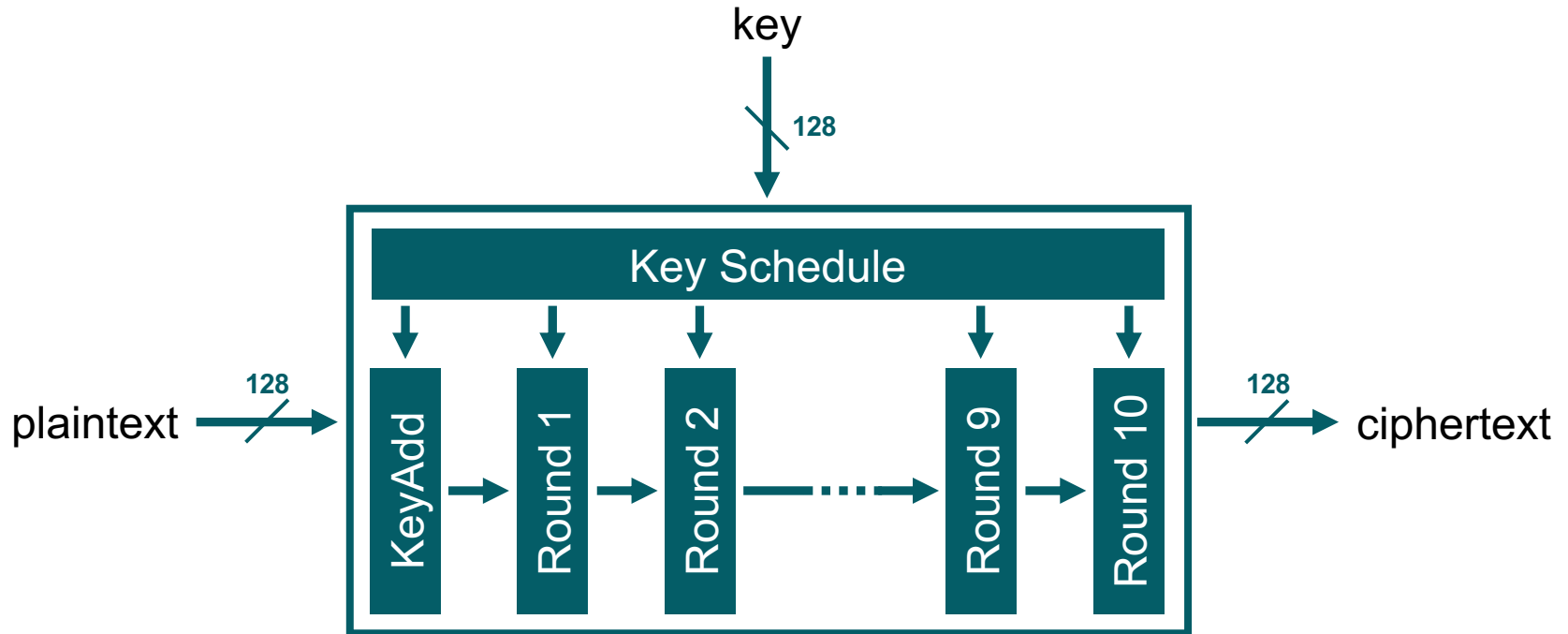
At the bottom of the video player, the following text is visible:

Lecture 8: Advanced Encryption Standard (AES) by Christof Paar
Introduction to Cryptography by Christof Paar
60.7K subscribers
2.7K
Share
Save

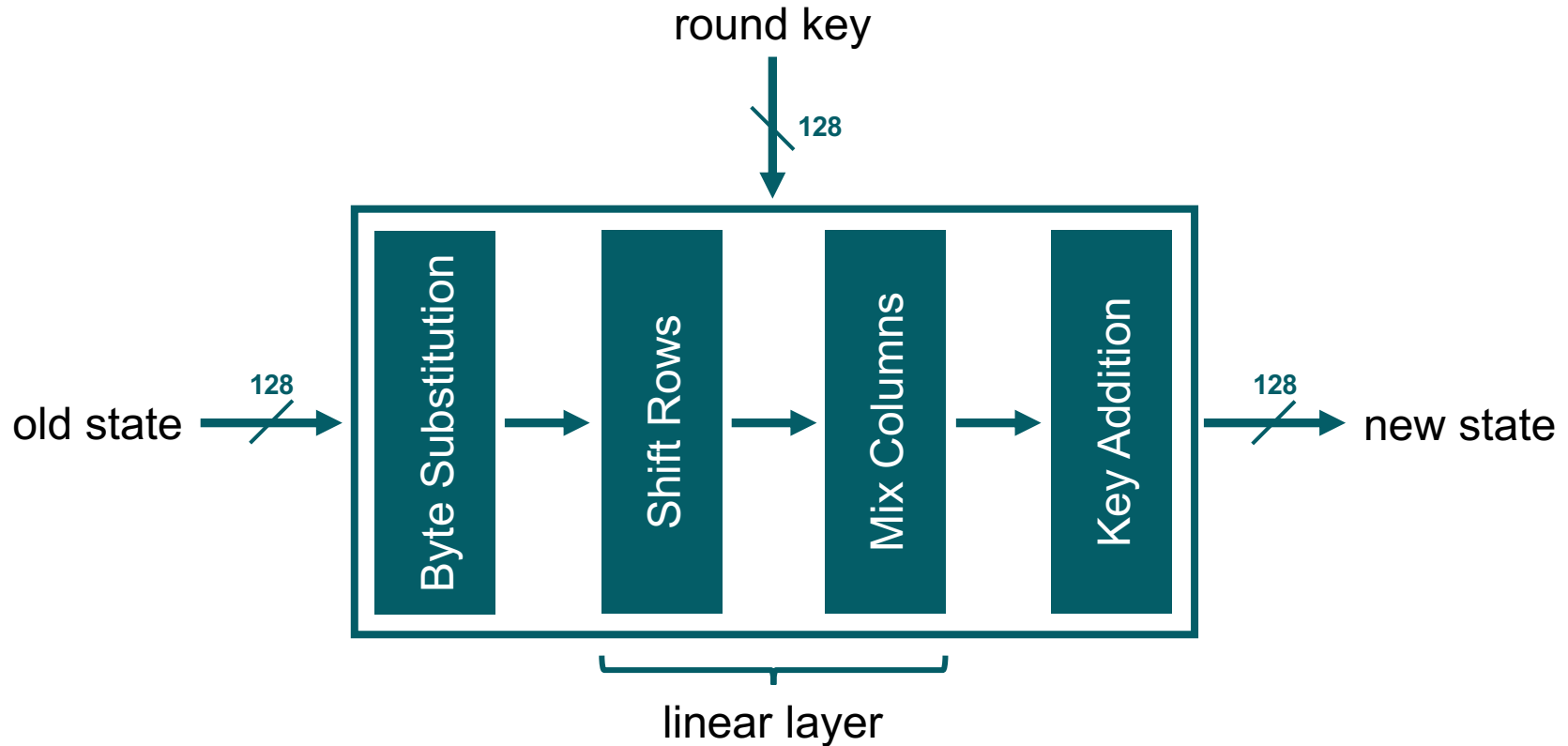
AES



AES

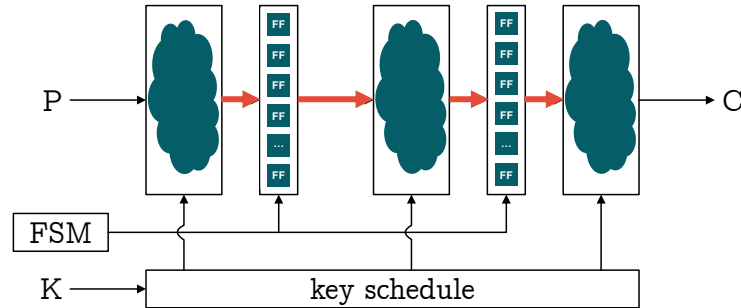


AES

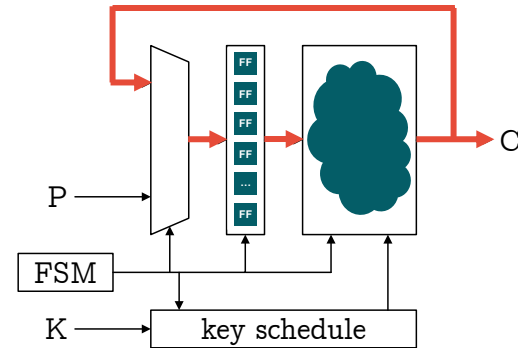


SYMMETRIC CRYPTOGRAPHY IN HARDWARE

Pipelined Implementation

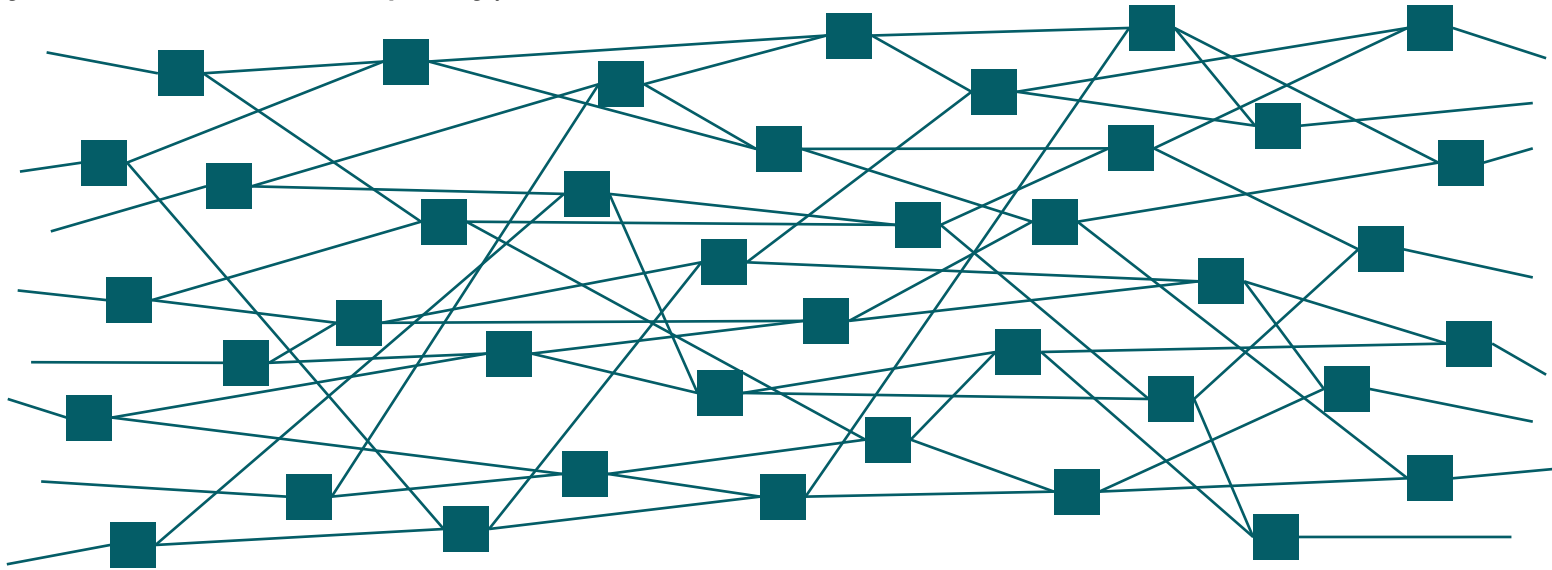


Round-Based Implementation

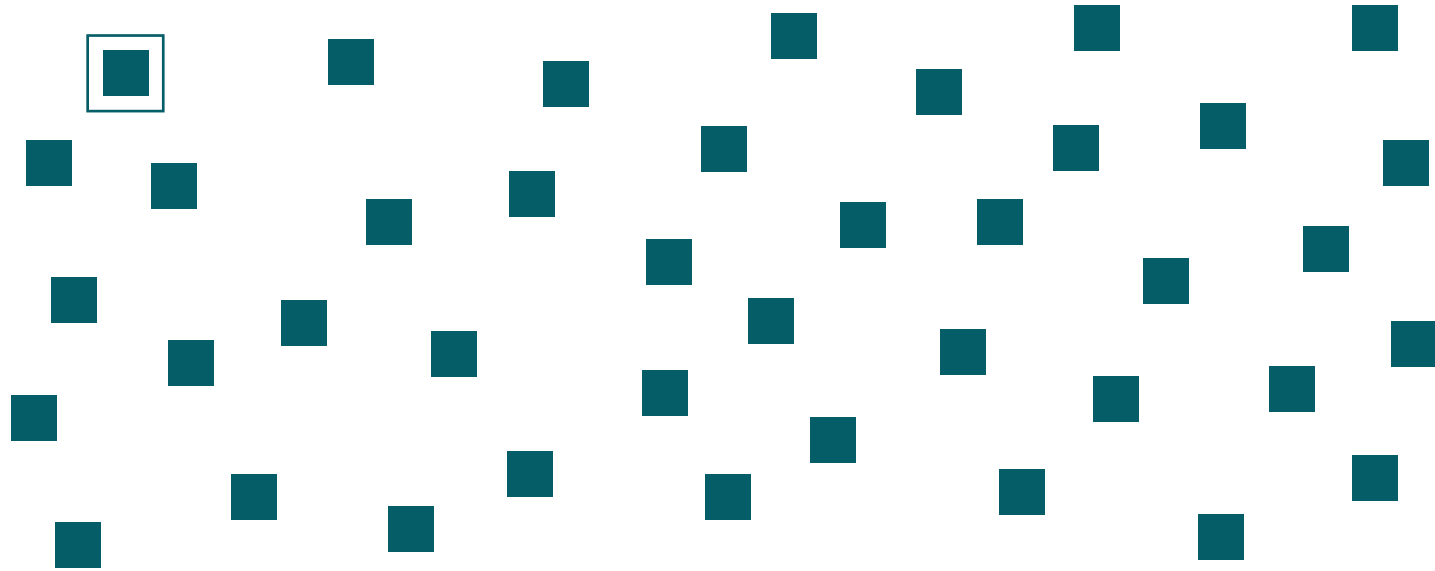


SYMMETRIC CRYPTOGRAPHY IN HARDWARE

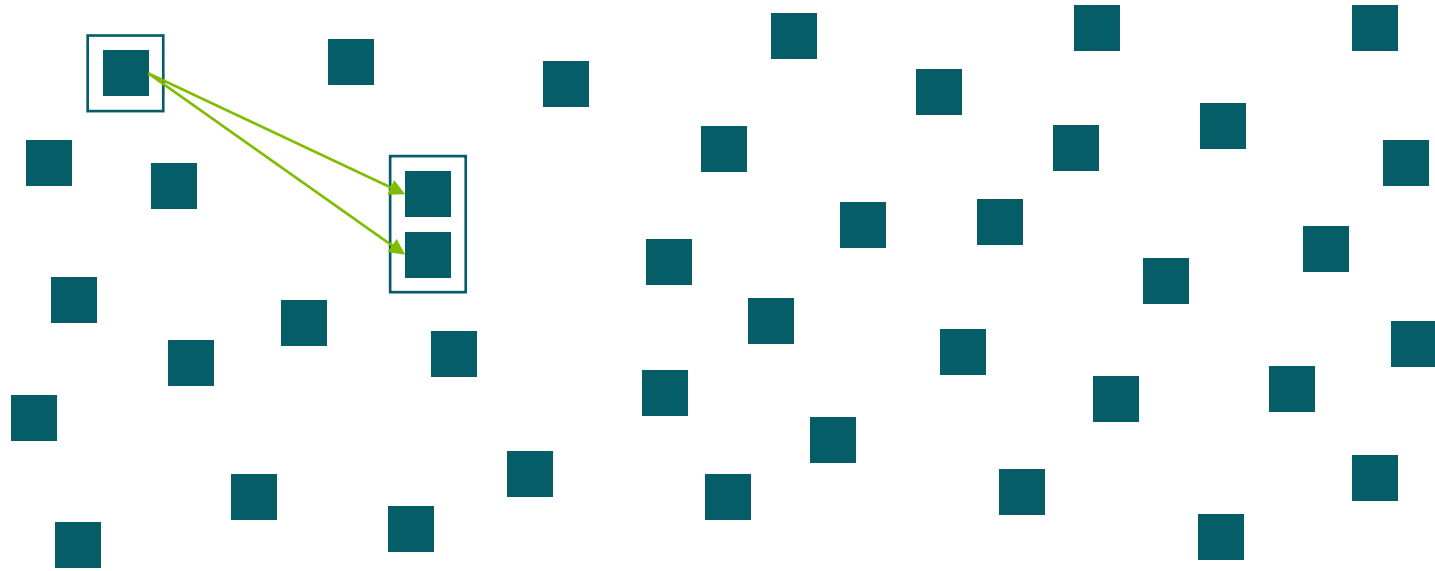
How it looks like as a netlist...
(only shows FFs for simplicity)



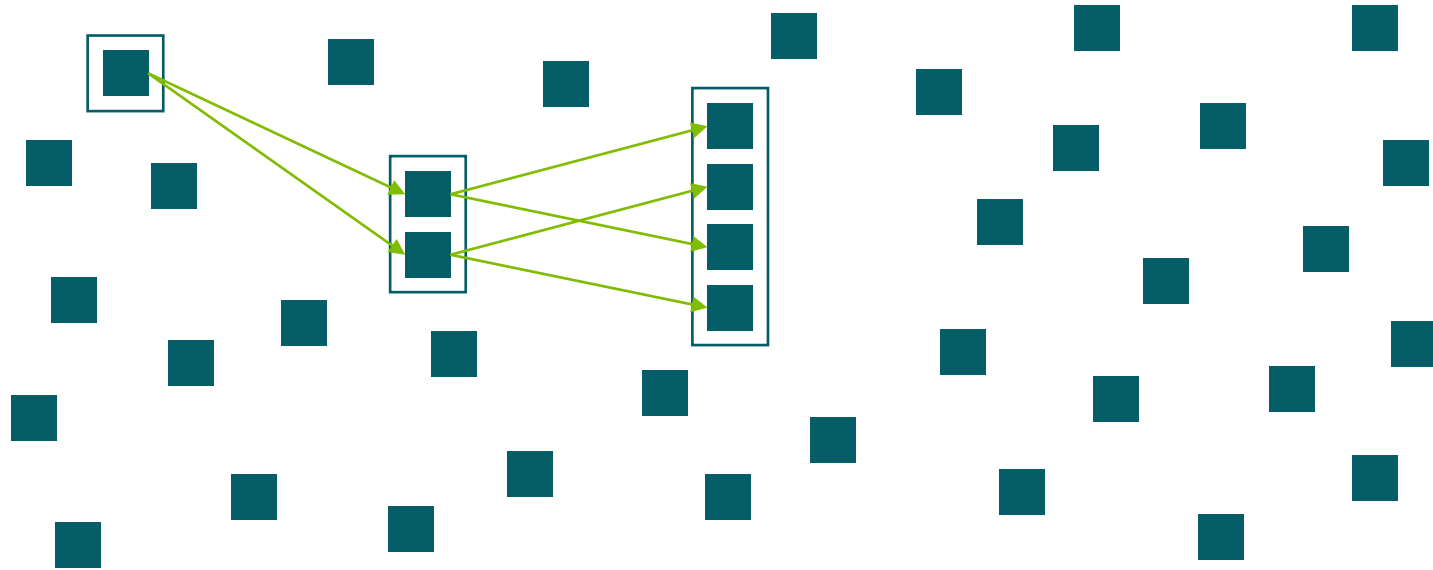
HAWKEYE: CANDIDATE SEARCH



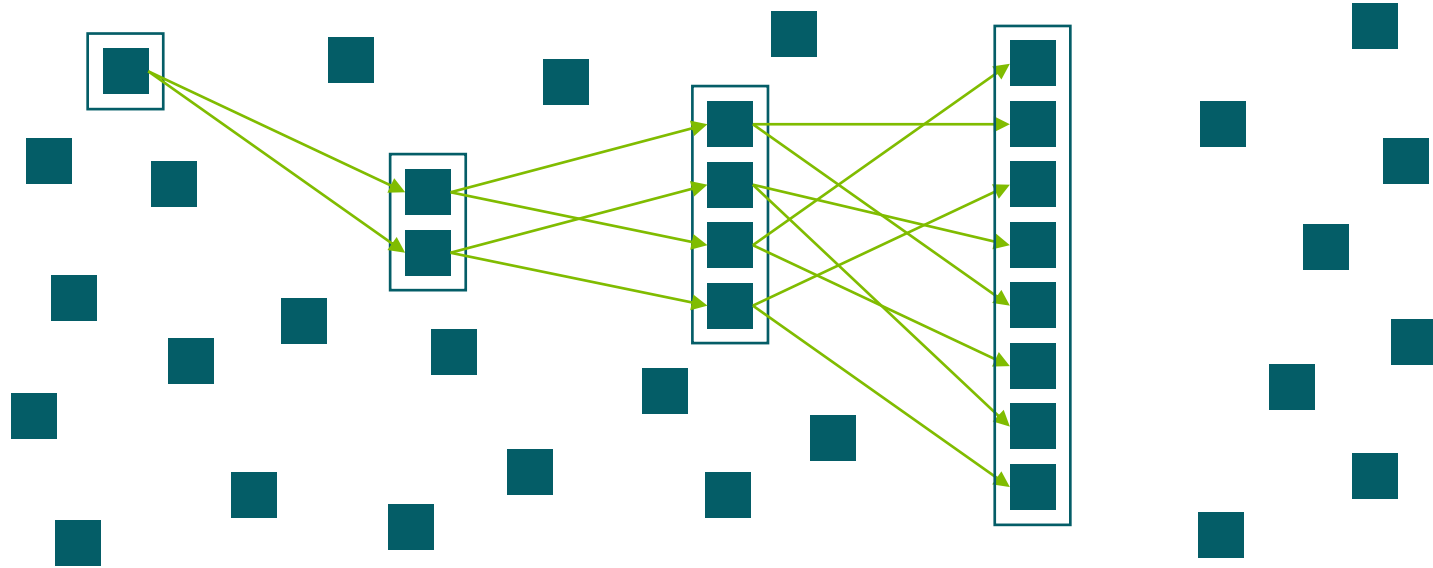
HAWKEYE: CANDIDATE SEARCH



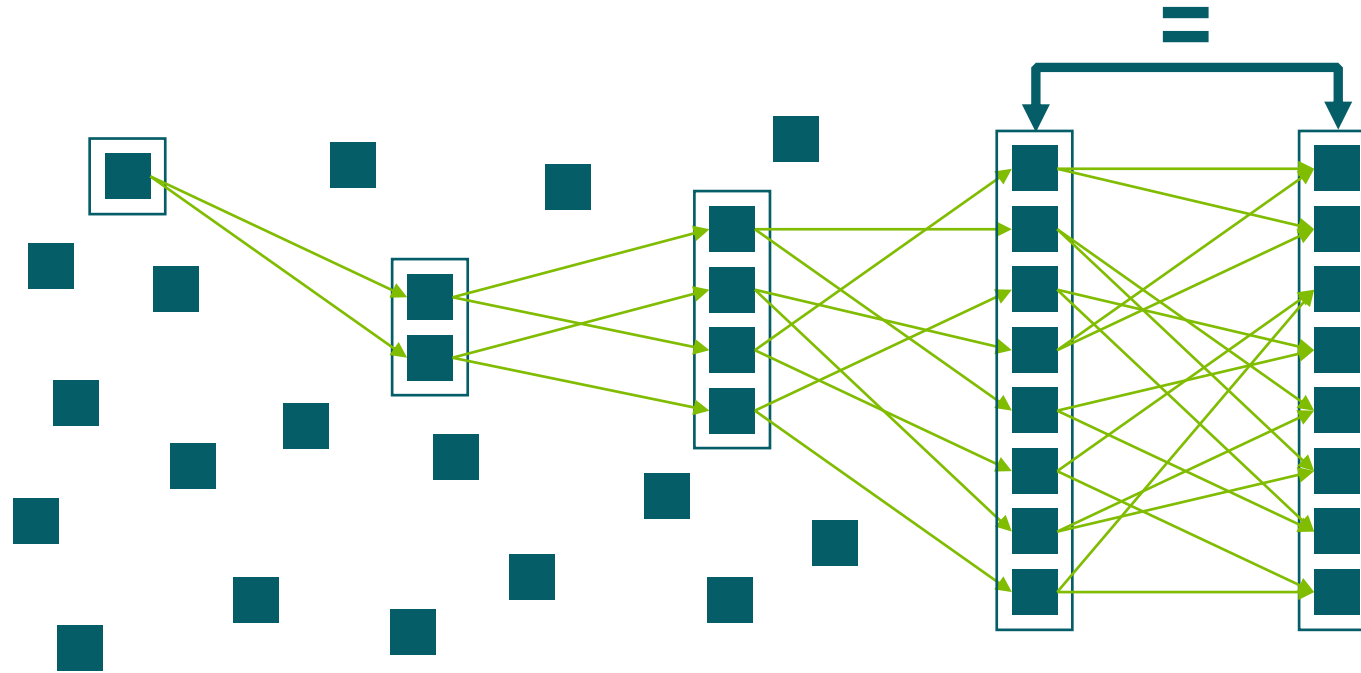
HAWKEYE: CANDIDATE SEARCH



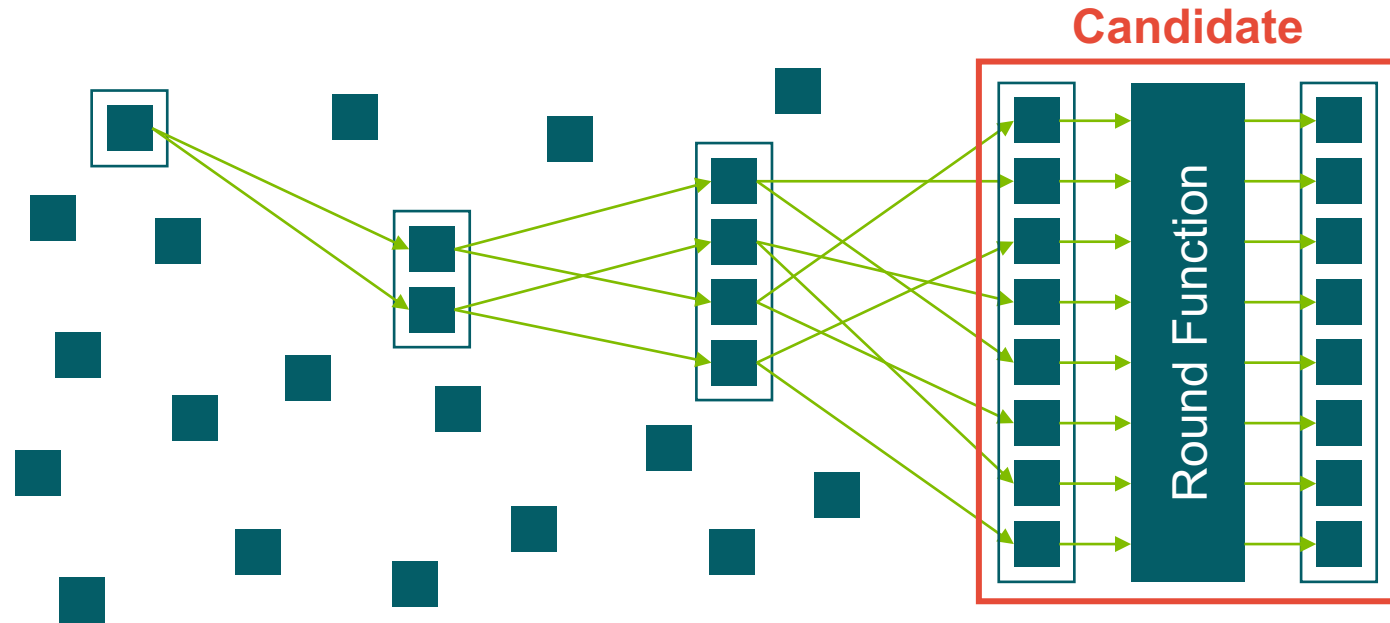
HAWKEYE: CANDIDATE SEARCH



HAWKEYE: CANDIDATE SEARCH

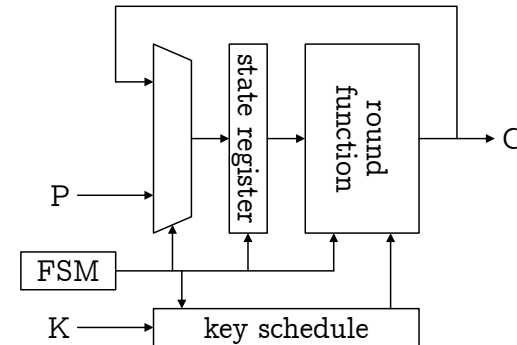
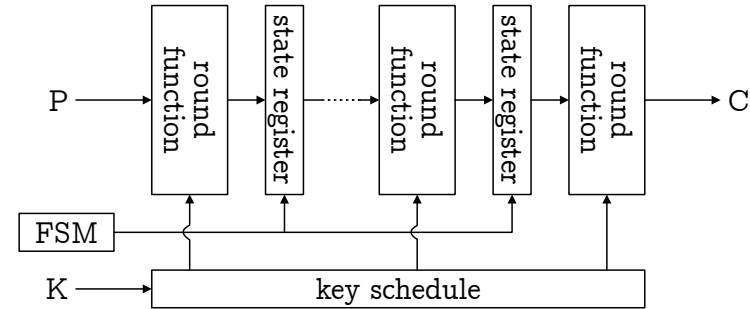


HAWKEYE: CANDIDATE SEARCH

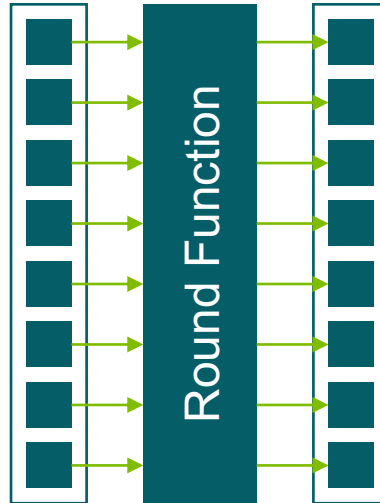


IMPORTANT OBSERVATIONS

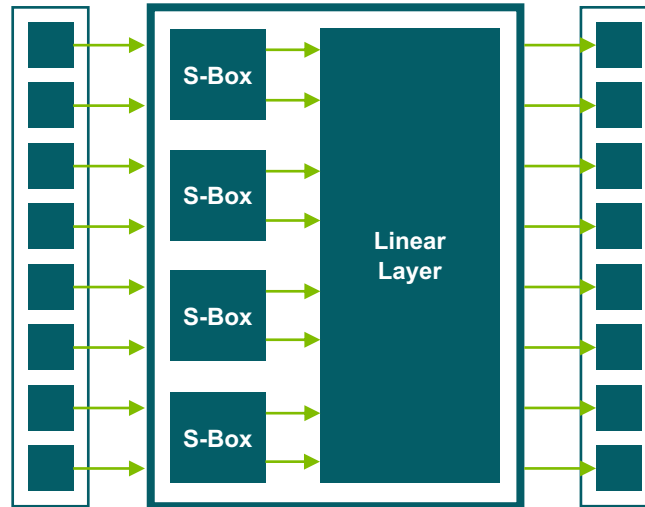
1. FFs in state register influence only state register and ciphertext output
2. State register FFs and ciphertext FFs are distinguishable
3. Avalanche effect: FF in first state register influences all FFs of later state register(s)
4. Round function only depends on plaintext, round keys, and FSM control signals



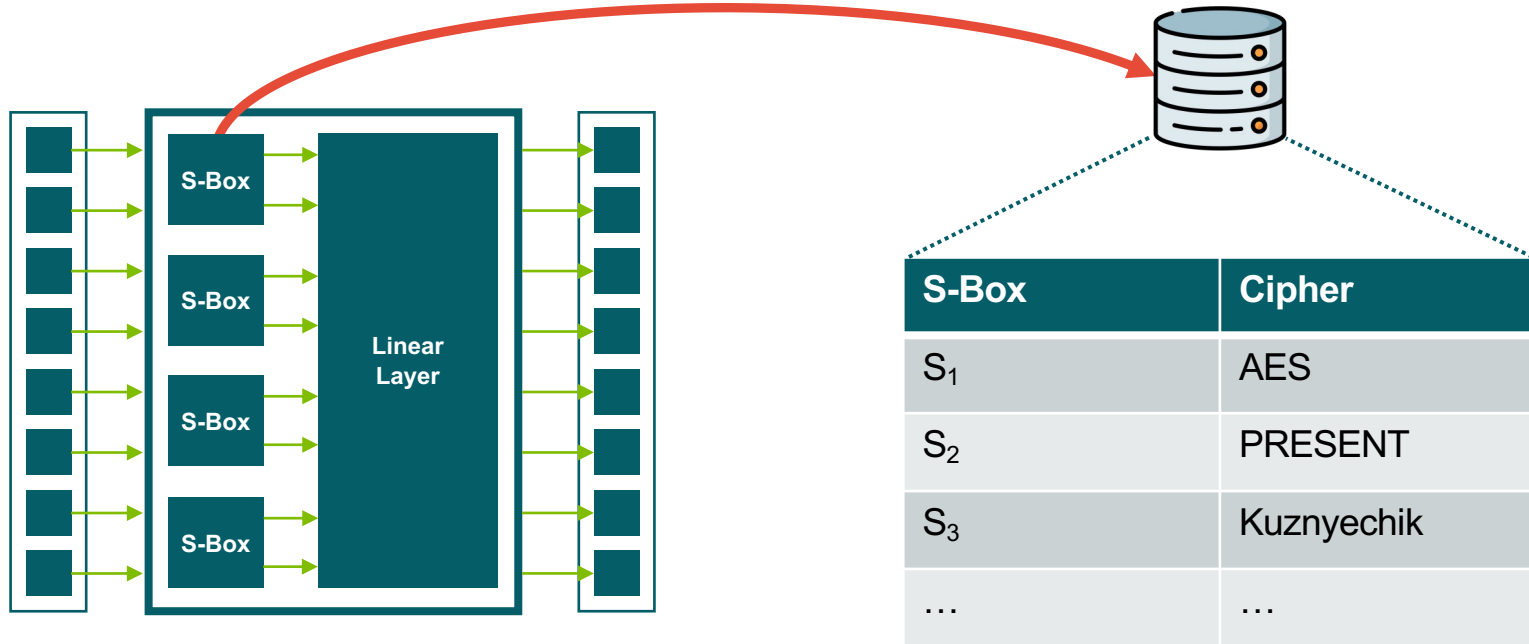
HAWKEYE: ROUND FUNCTION ANALYSIS



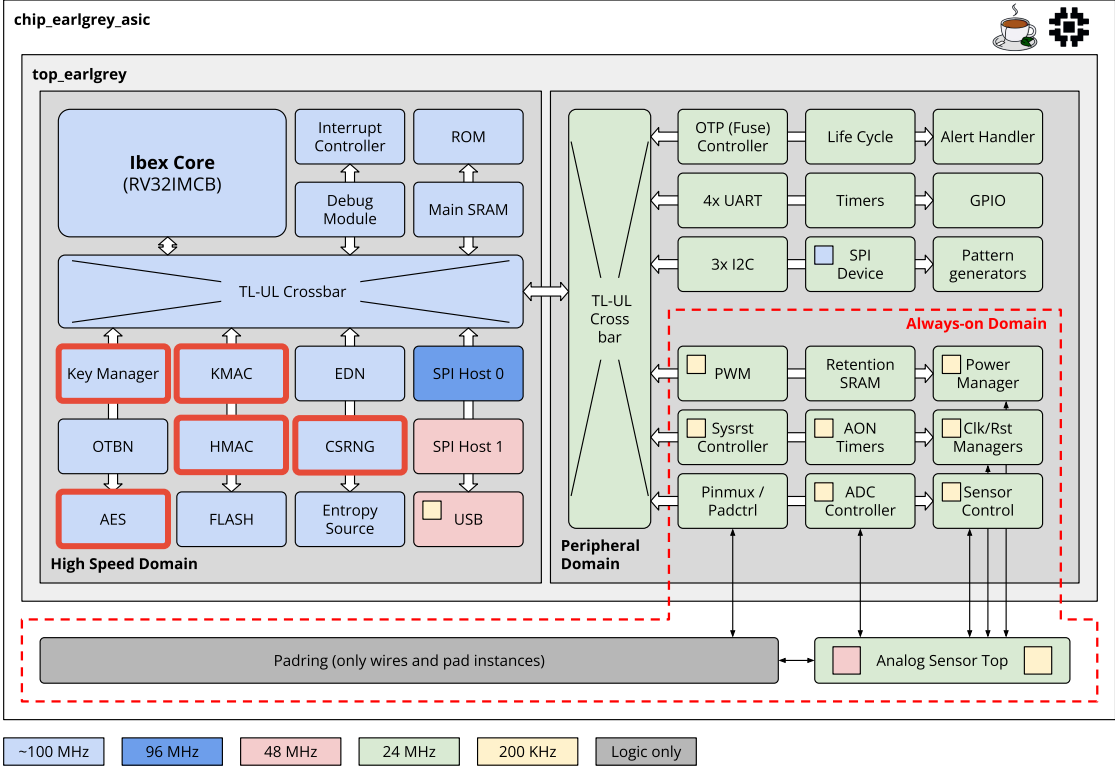
HAWKEYE: ROUND FUNCTION ANALYSIS



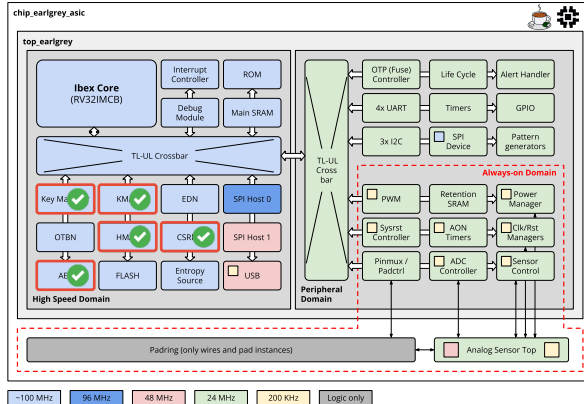
HAWKEYE: ROUND FUNCTION ANALYSIS



EVALUATION: OPENTITAN



EVALUATION: OPENTITAN



Contains 424.341 gates

After 77.3 seconds:

No.	#FFs	Crypto?	Description
1	640	✓	partial Keccak state
2	128	✓	AES state
3	256	✓	AES round key
4	256	✓	SHA-2 state
5	256	✓	Xoshiro256++ state
6	192	✓	PRESENT state and key
7	64	✓	PRINCE output
8	64	✓	LFSR of PRNG within analog sensors
9	64	✓	key manager clearing PRNG
10	64	✓	AES clearing PRNG
11	40	✓	LFSR of PRNG in memory controller
12	40	✓	LFSR of PRNG in memory controller

EVALUATION: KNOWN CIPHERS

Name	Suc.	FPGA						ASIC						Gates	Src.				
		Method 1			Method 2			S?	Method 1			Method 2				S?			
		TP	FP	t	TP	FP	t		TP	FP	t	TP	FP	t					
3DES	✓	3	0	0.01s	3	0	1.75s	✗	3458	✓	0	2	0.35s	2	0	1.33s	✗	9916	[46]
AES-128 _r	✓	1	0	0.11s	2	0	2.74s	✓	3458	✓	0	1	1.34s	1	0	2.92s	✓	12804	[1]
AES-128 _p	✗	0	0	0.01s	0	0	0.24s	✗	8064	✓	7	0	1.10s	0	0	24.86s	✗	154477	[2]
ASCON	✓	0	0	0.03s	1	0	12.94s	✓	5323	✓	0	1	4.73s	1	0	14.14s	✗	11786	[48]
CRAFT	✓	1	0	0.01s	1	0	0.93s	✓	1547	✓	0	1	0.34s	1	0	1.19s	✓	3503	[1]
DES	✓	13	0	0.07s	0	0	2.19s	✗	3956	✓	13	0	0.11s	0	0	2.24s	✗	19976	[2]
GIFT	✓	1	0	0.01s	1	0	0.43s	✓	1512	✓	0	1	0.21s	1	0	0.61s	(✓)	2969	[1]
LED-64	✓	1	0	0.01s	1	0	0.35s	✓	1328	✓	0	1	0.22s	1	0	0.48s	✓	2798	[1]
LED-128	✓	1	0	0.01s	1	0	0.49s	✓	1541	✓	0	1	0.30s	1	0	0.64s	✓	2989	[1]
Magma	✓	1	0	0.01s	1	0	1.07s	✗	1808	✓	1	0	0.29s	1	0	0.90s	✗	4894	[46]
Midori	✓	1	0	0.01s	1	0	0.67s	✓	1489	✓	0	1	0.44s	1	0	0.86s	✗	2824	[1]
Piccolo	✓	1	0	0.01s	1	0	0.62s	✗	1405	✓	0	1	0.41s	1	0	0.78s	✗	3500	[1]
PRESENT-80	✓	1	0	0.01s	2	0	0.35s	✓	1302	✓	1	0	0.06s	2	0	0.25s	✓	2937	[46]
PRESENT-128	✓	1	0	0.01s	2	0	0.47s	✓	1538	✓	0	1	0.27s	2	0	0.66s	(✓)	3486	[1]
SHA-256	✓	1	0	0.27s	1	0	19.98s	-	3547	✓	0	2	3.37s	1	0	21.38s	-	7152	[46]
SHA-3	✓	1	0	7.06s	1	0	87.81s	✗	8143	✓	0	1	42.68s	1	0	93.32s	✗	22414	[46]
SIMON-128	✓	1	0	0.02s	1	0	0.28s	-	1539	✓	0	1	0.21s	1	0	0.47s	-	3620	[1]
SKINNY-64	✓	1	0	0.01s	1	0	0.52s	✓	1221	✓	0	1	0.19s	1	0	0.65s	✓	2197	[1]
KATAN-80	✓	0	0	0.00s	1	0	0.14s	-	1246	✓	0	1	0.10s	1	0	0.30s	-	3160	[1]
KTANTAN-80	✓	0	0	0.00s	1	0	0.15s	-	1118	✓	0	1	0.09s	1	0	0.31s	-	2543	[1]
PRINCE	✗	0	0	0.05s	0	0	0.68s	✗	2600	✗	0	193	0.34s	0	0	1.42s	✗	7238	[29]
RSA-512	✗	0	4	13.77s	-	-	-	-	20908	✗	0	3	45.98s	-	-	-	-	82167	[2]
Trivium ₈	✗	0	0	0.01s	0	1	0.12s	-	1225	✓	0	0	0.07s	1	0	0.24s	-	3022	[19]
Trivium ₃₂	✓	0	0	0.01s	1	0	0.71s	-	1427	✗	0	0	0.13s	0	0	0.69s	-	2867	[19]
Trivium ₆₄	✓	0	1	0.01s	1	0	2.32s	-	1683	✗	0	2	0.27s	0	1	1.45s	-	3532	[19]
Ethernet	✓	-	0	0.10s	-	0	12.27s	-	5676	✗	-	1	1.46s	-	-	-	-	42161	[46]
CIC filter	✓	-	0	0.01s	-	0	0.09s	-	572	✓	-	0	0.06s	-	0	0.20s	-	1221	[46]
Hilbert trans.	✓	-	0	0.02s	-	0	0.27s	-	1390	✓	-	1	0.09s	-	0	0.39s	-	3004	[46]
CPU Edge	✗	-	0	0.59s	-	2	428.20s	-	11144	✗	-	3	2.53s	-	2	156.77s	-	41909	[2]
CPU Ibex	✗	-	1	0.16s	-	1	80.81s	-	6340	✗	-	1	0.89s	-	-	-	-	12751	[2]
CPU open8	✗	-	0	0.01s	-	1	0.80s	-	1216	✗	-	0	0.02s	-	1	0.77s	-	1888	[2]

- Finds almost all ciphers in FPGA and ASIC netlists
- Runtime is in the seconds
- Even finds some ciphers that we did not expect it to find
- Only very few false positives

CONCLUSION & FUTURE WORK

Recap:

- HAWKEYE is optimized for SPN, ARX, and Feistel ciphers
- It reliably locates all kinds of ciphers, even in a large industry-grade SoC
- Detection is fast and usually requires at most a few minutes

Future Work / Please Reach Out:

- Symmetric cryptography based on shift registers
- Side-channel protected implementations
- Actually finding unknown cryptography
- **If you have a real-world device to look at, please reach out to us!**

THANK YOU

CASA

CYBER SECURITY IN THE AGE
OF LARGE-SCALE ADVERSARIES

RUHR-UNIVERSITÄT BOCHUM
Horst-Görtz-Institut für IT-Sicherheit
Exzellenzcluster CASA

MC 0.75 | Universitätsstr. 150 | 44780 Bochum | Germany
www.casa.rub.de | www.hgi.rub.de

Gefördert durch

DFG Deutsche
Forschungsgemeinschaft



RUHR
UNIVERSITÄT
BOCHUM

RUB