IMSE -cnm Instituto de Microelectrónica de Sevilla | CSIC Consejo Superior de Investigaciones Científicas | Universidad de Sevilla

Design of a hardware Root-of-Trust on embedded systems

Eros Camacho-Ruiz

Supervisors:
Dr. Piedad Brox Jiménez
Prof. Francisco Vidal Fernández Fernández

Sevilla, March 13th 2024

Introduction

Cybersecurity

Information security

- Critical Infraestructure
- Data privacy preserving
- Economic and National Security Implications
- Global Communications

NIST National Institute of Standards and Technology U.S. Department of Commerce

enisa EUROPEAN UNION AGENCY FOR CYBERSECURITY

CCN centro criptológico nacional

Introduction

Hypothesis

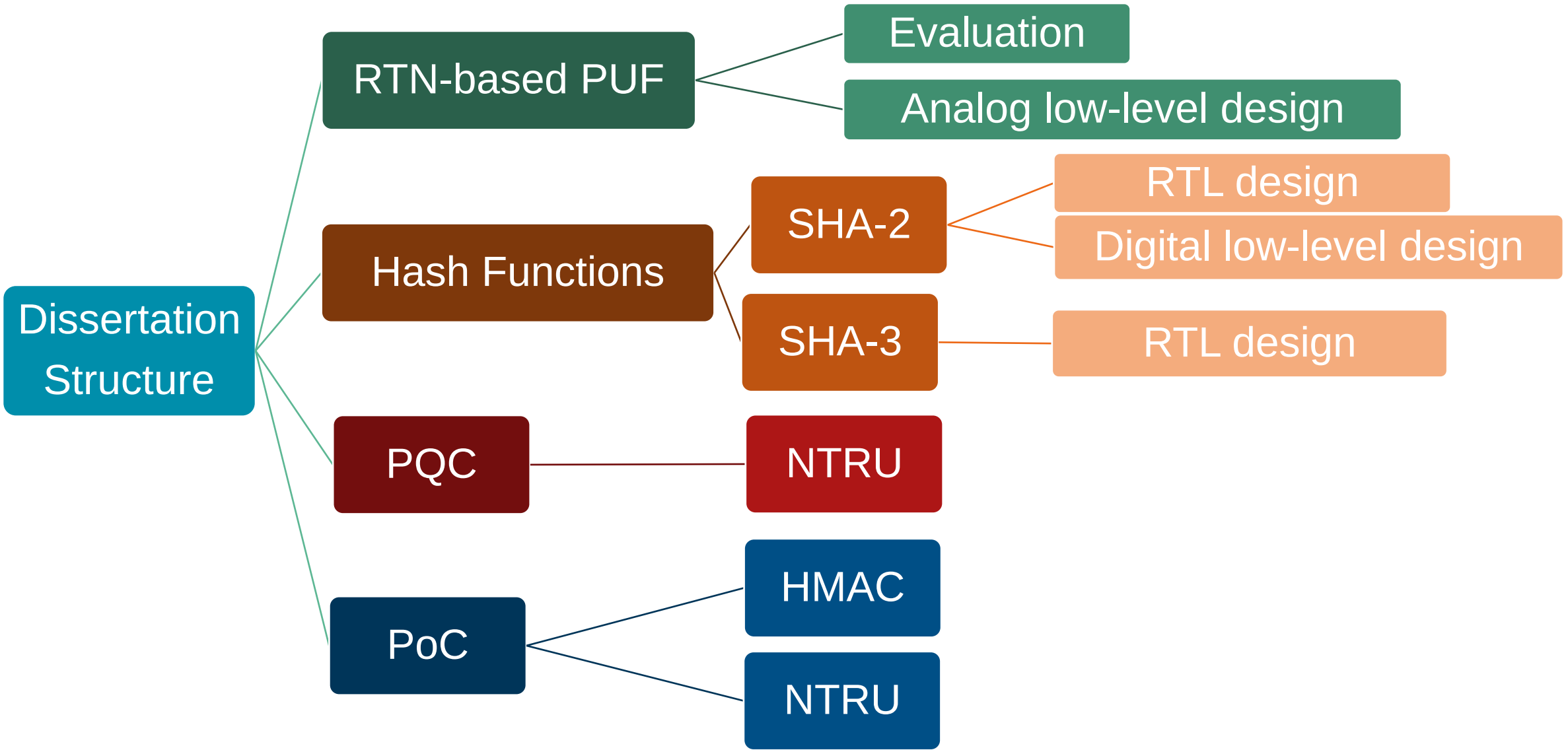
Contributing to improve the security of information using hardware solutions in IoT environments where restrictions are usually stronger.

How ?

Proposing a Root-of-Trust composed of different modules that have been designed separately.

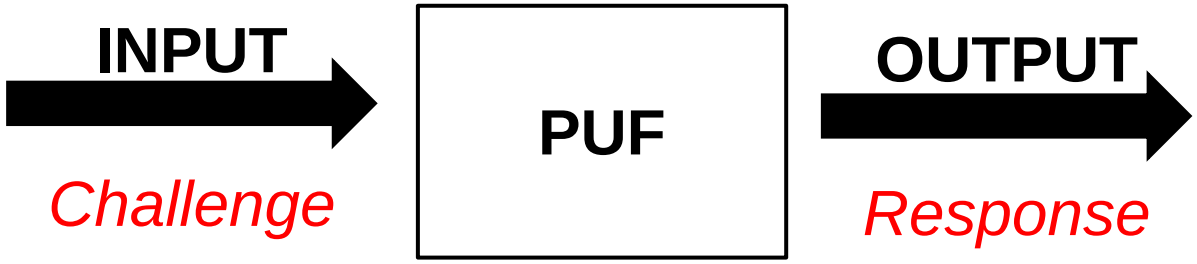
Use Cases to demonstrate the feasibility of the dissertation

Overview

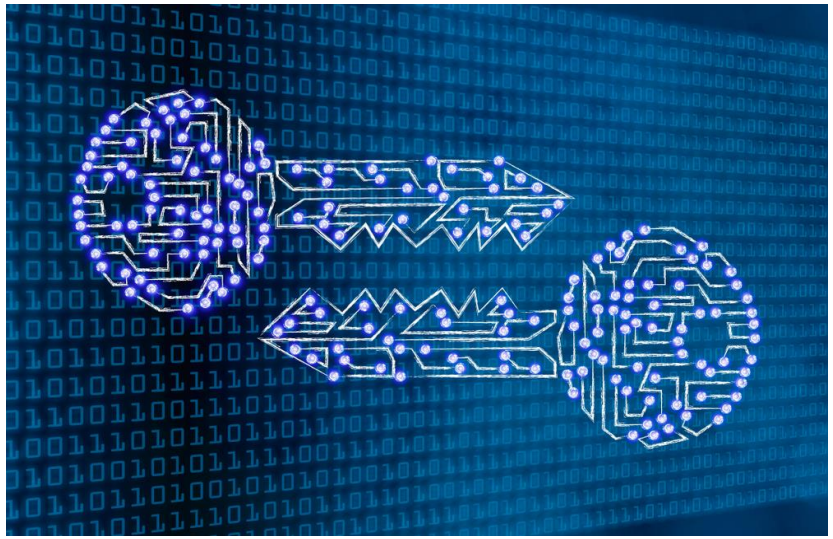

Dissertation Structure
RTN-based PUF
Evaluation
Analog low-level design
Hash Functions
SHA-2
RTL design
Digital low-level design
SHA-3
RTL design
PQC
NTRU
PoC
HMAC
NTRU

What is a PUF ?

Physical Unclonable Function
(PUF)



Key Obfuscation

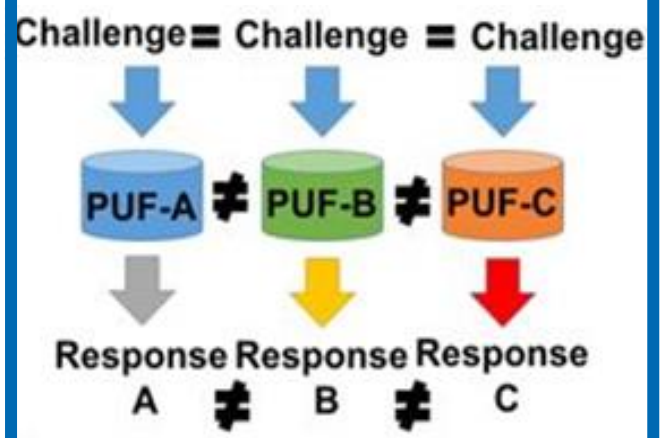


Device identification

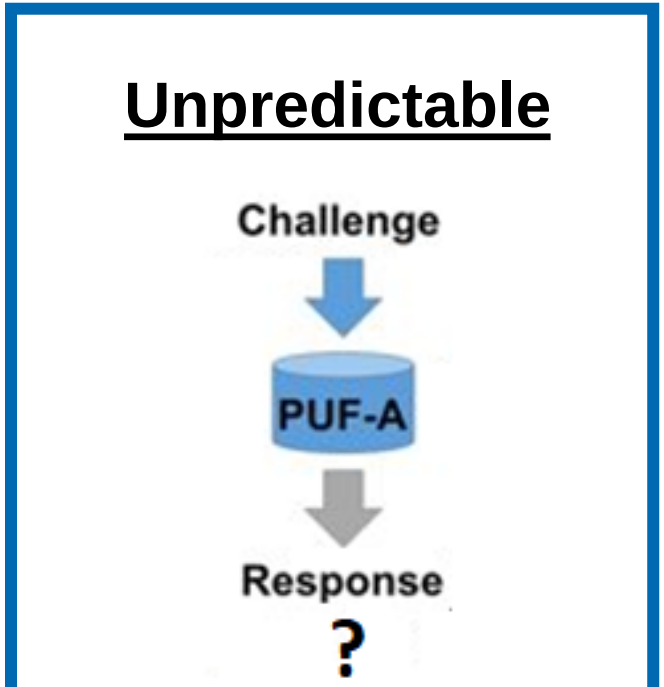
What is a PUF ?

Response Characteristics

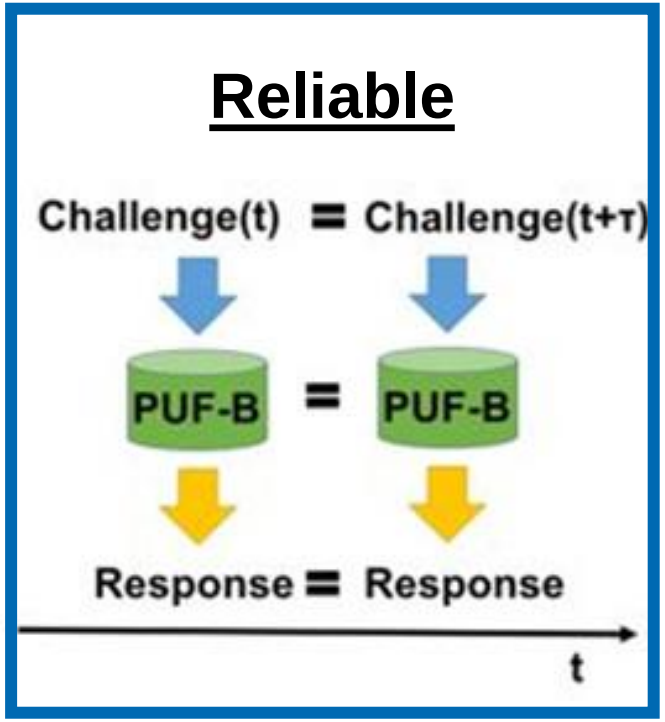
Unique



Unpredictable

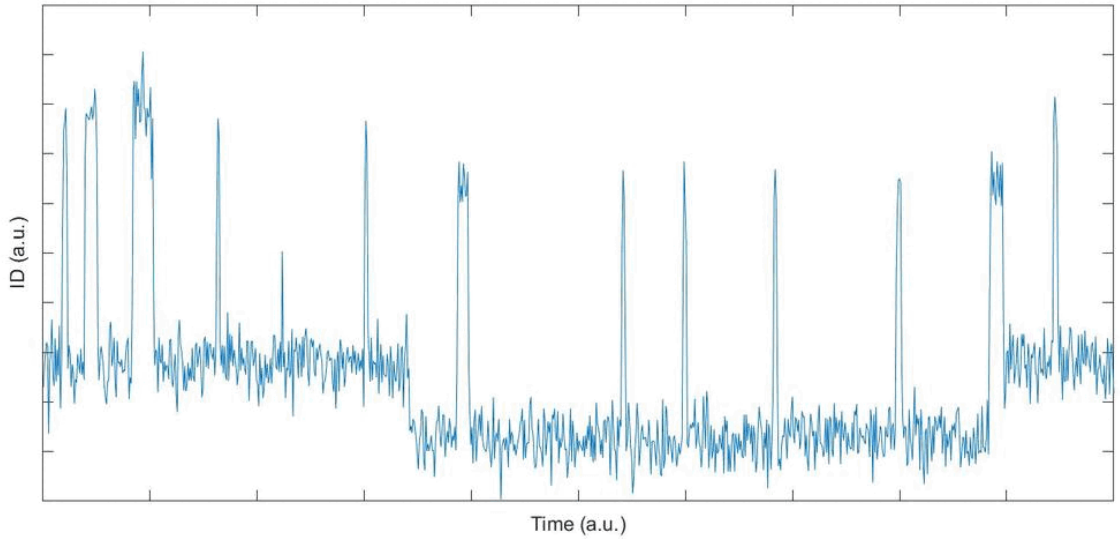


Reliable



A novel RTN-based PUF

Patent: WO/2023/202843
MÉTODO Y DISPOSITIVO PARA FUNCIÓN FÍSICA NO CLONABLE (PUF) BASADA EN RUIDO TELEGRÁFICO ALEATORIO (RTN)



Random Telegraph Noise (RTN)

Low biasing voltages → Prevent aging

Handle the RTN

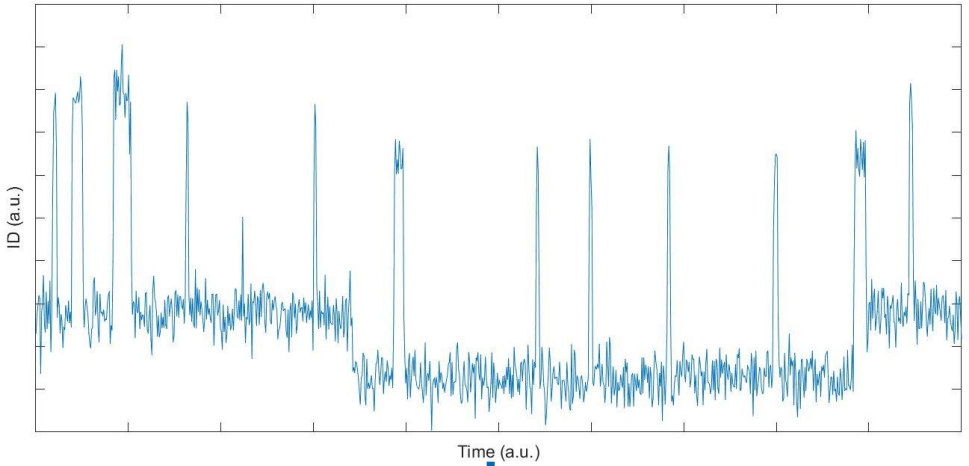
Maximum Parameter Fluctuation (MPF)

<u>Cumulative</u> difference between the maximum and the minimum in a time interval of a drain CURRENT / VOLTAGE trace

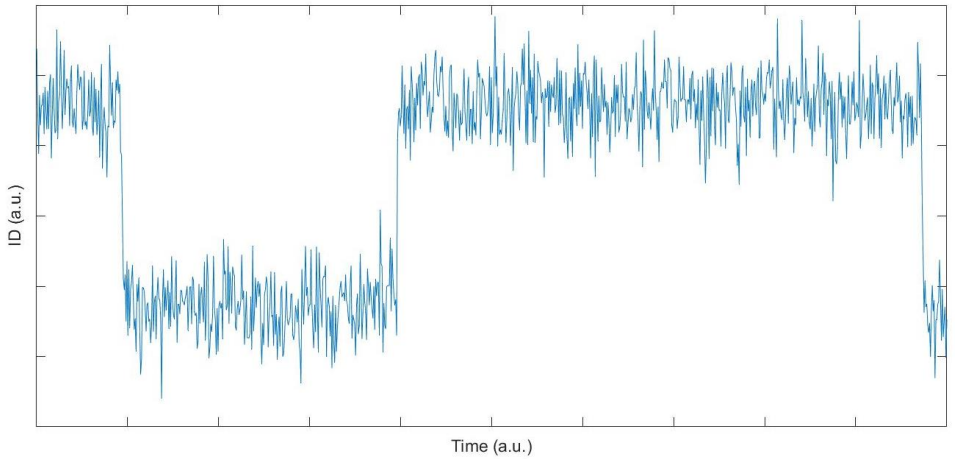
A novel RTN-based PUF: Schematic

Pair of transistors

Current/Voltage trace with RTN



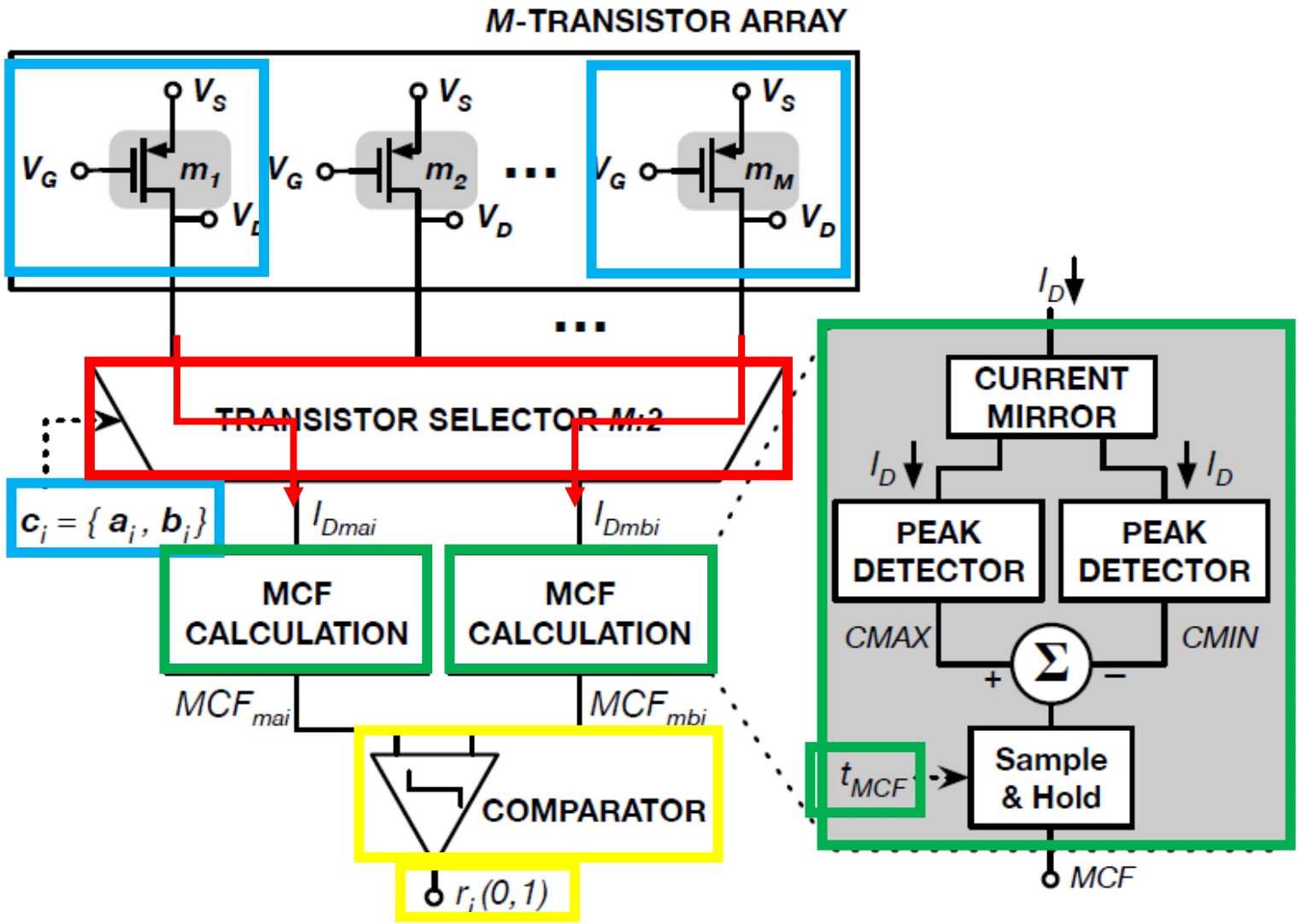
Current/Voltage trace with RTN

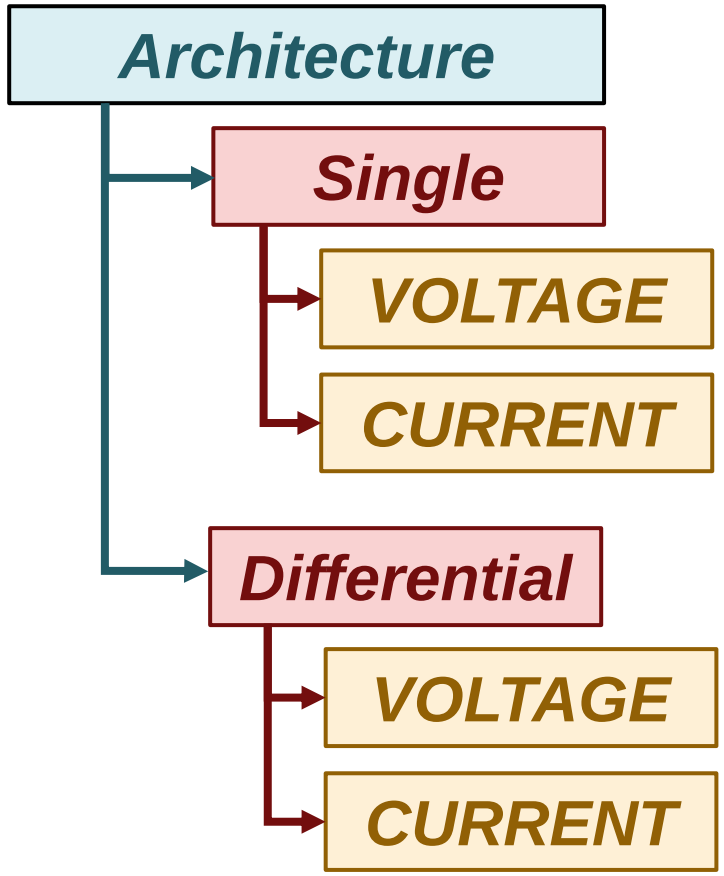


MPF → **Comparison** ← **MPF**

1-bit Response = {0,1}

A novel RTN-based PUF: Architecture





A novel RTN-based PUF: Metrics

Probability

$$P = max\left(n_0 / L, n_1 / L \right)$$

Number of zeros/ones → n_0 / n_1
Number of measurements → L

Bit Selection Method

Metrics

- **Probability (P)**
- **Reliability (1-HDIntra)**
- **Uniqueness (HDInter)**
- **Unpredictability (HW)**

A novel RTN-based PUF: Evaluation

Evaluation

Low-level design

Is it feasible to build a PUF based on RTN?

How sizes and biasing conditions can alter the PUF performance?

How non-idealities of the circuitry needed to get the response can affect the PUF performance?

Differential Current mode

A novel RTN-based PUF: Evaluation #1

Is it possible to obtain a reliable response from this PUF?

Comparing with similar area PUFs

32-bit response

	[7] Mispan, IEEE Prime 2015	[8] Mispan, IOLTS 2016	[9] Zhuang, IEEE TCAS I, 2020	This diss.
Reliability (%)	91.58	-	94.2	**99.8**
HD_{inter} (%)	50.23	49.9	49.9	**49.99**
HW (%)	-	53.94	53.94	**50.09**

Comparing with similar quality metrics PUFs

128-bit response

	[14] Yoshinaga ISCAS 2016	[12] Santana-Andreo, Integration, 2022	This diss.
Reliability (%)	99.99	99.99	**99.99**
HD_{inter} (%)	48.0	-	**49.97**
HW (%)	-	50	**49.96**
Technology	65-nm	65-nm	65-nm
Number of transistors	34,816 (2,048 17T-ROs)	4,992 (832 6T-SRAM cells)	**2,286**

A novel RTN-based PUF: Evaluation #2

How sizes and biasing conditions can alter the PUF performance?

↑ *Larger Transistors*
↑ *Stable Pairs*

↓ *Smaller Transistors*
↓ *Area Cost per Bit*

OPTIMUM BIASING
$|V_{GS}| = 0.8\ V$ & $|V_{DS}| = 1.2\ V$

OPTIMUM SIZE
80x60 nm²

A novel RTN-based PUF: Evaluation #3

How non-idealities of the circuitry needed to get the response can affect the PUF performance?

Non-idealities of the circuitry can affect the performance of the PUF

MCF Extractor and Comparator must be carefully designed

The on-resistance and off-resistance of the transistor selector can affect the performance

Low-level design of the RTN-based PUF

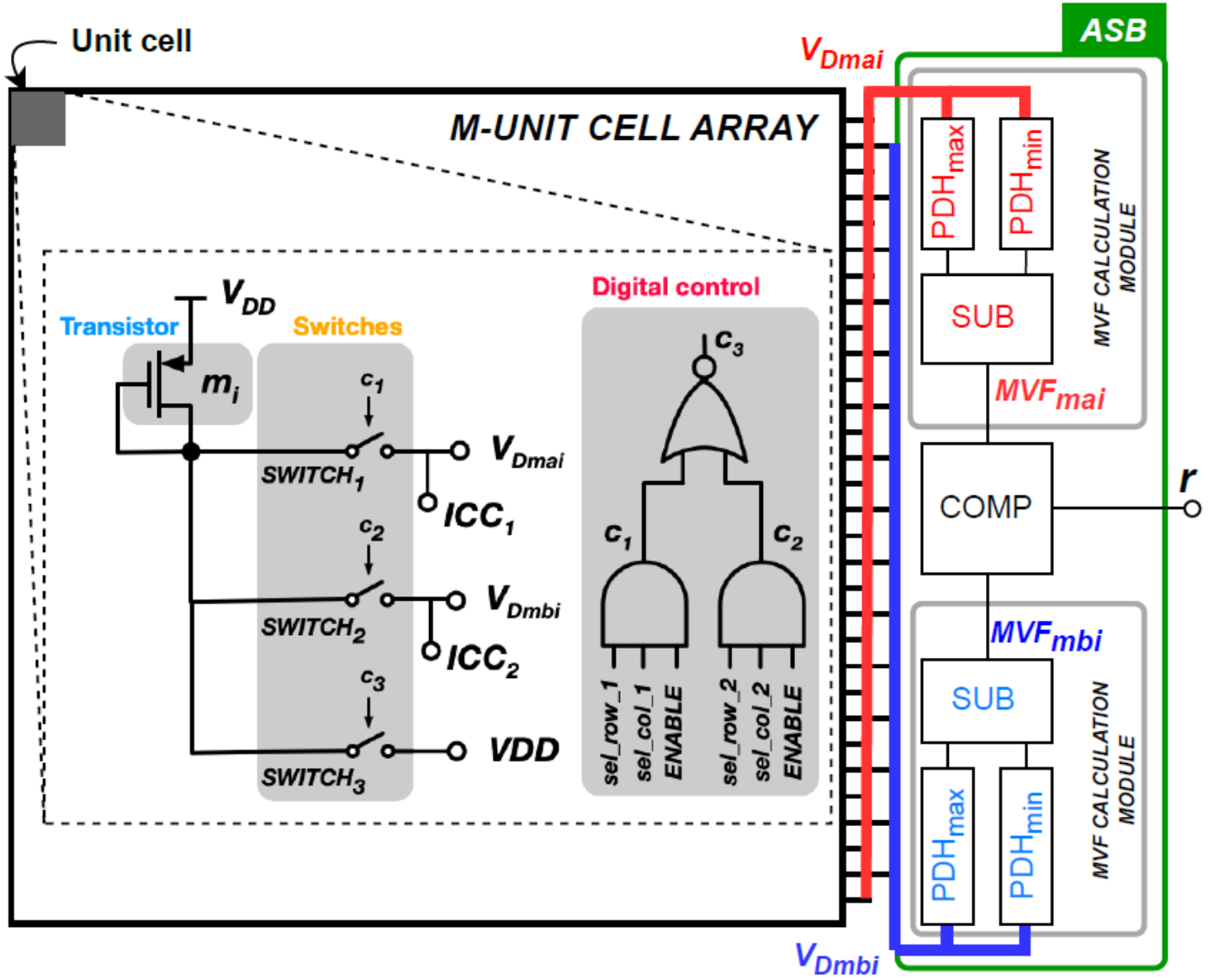
Evaluation → **Low-level design**

Analog design flow

Differential Voltage mode

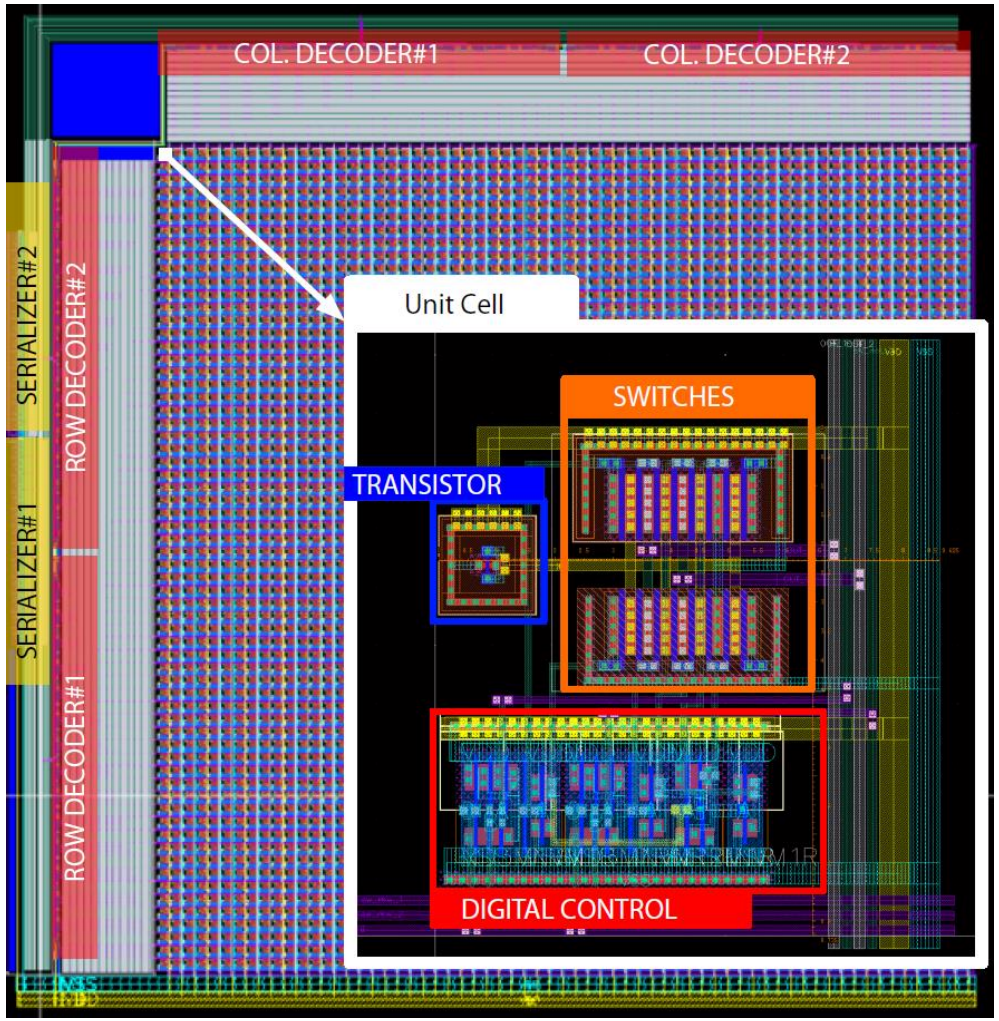
1. Avoiding potential issues in current sensing
2. Examination of different ICC in CC method can improve the PUF quality.
3. In case aging impacts the PUF, adjusting voltage levels is a simple task.

Low-level design: floorplan



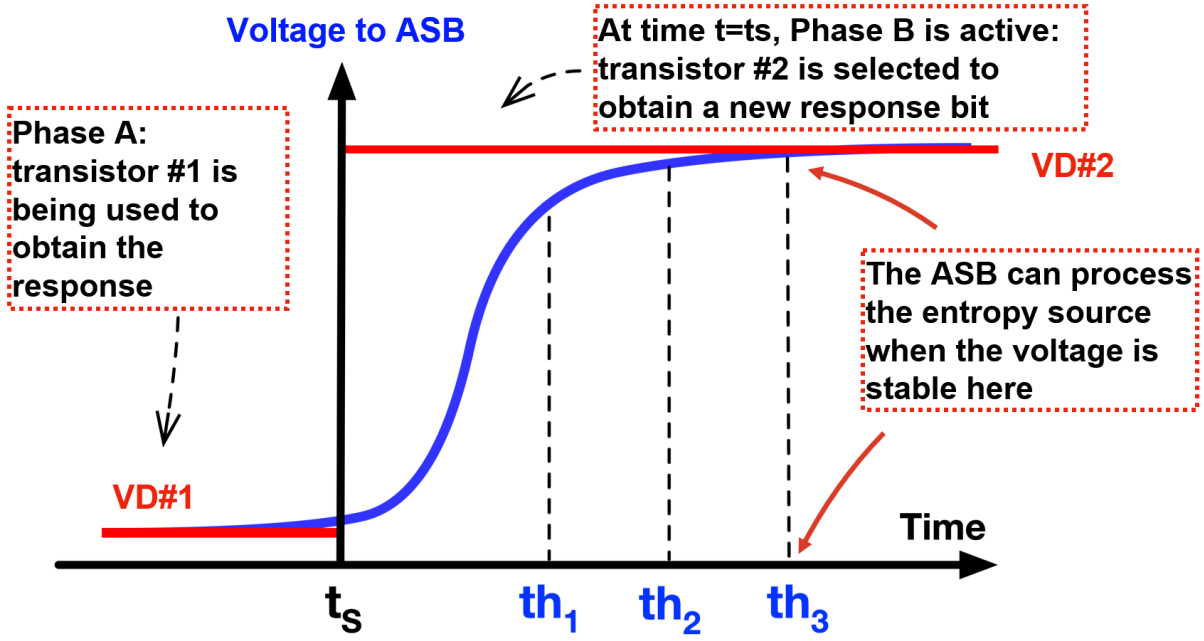
Low-level design: array design

4,096 transistors

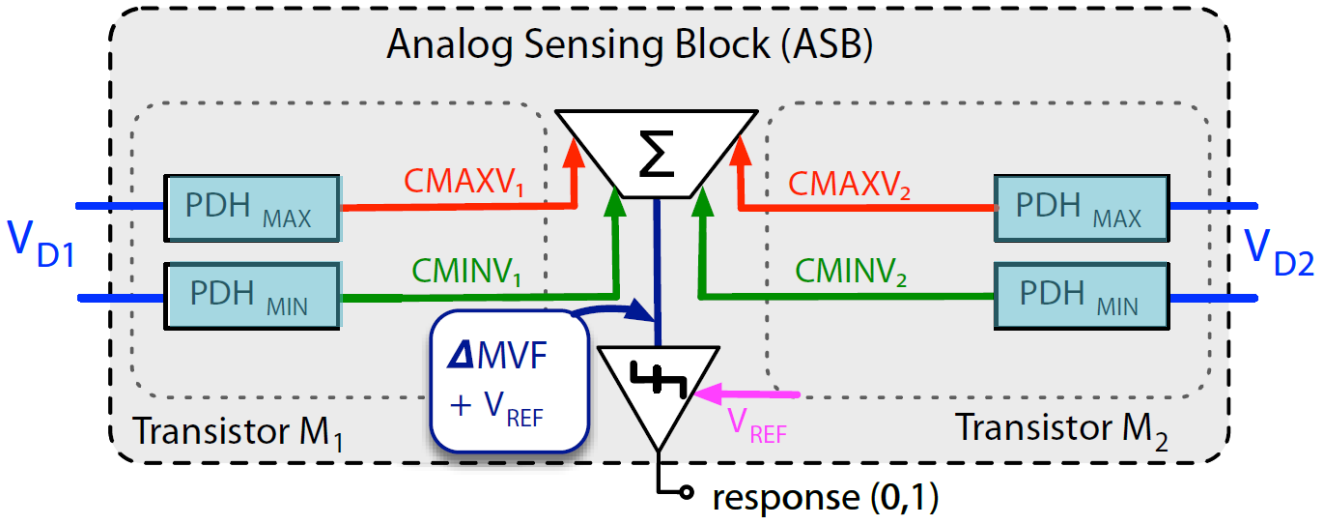


OPTIMUM SIZE
L = 100nm
W = 2um

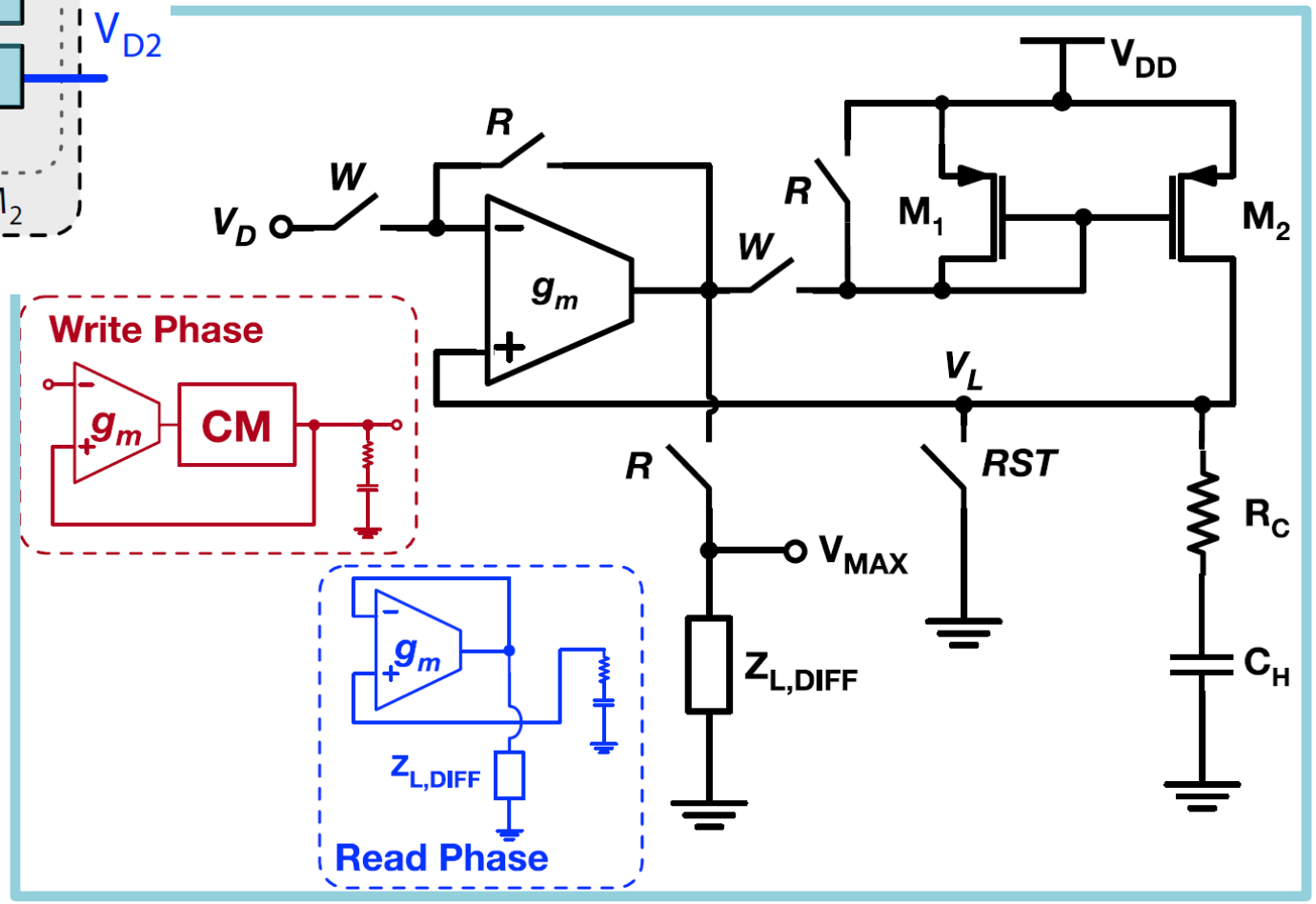
tmin = 10us



Low-level design: ASB design

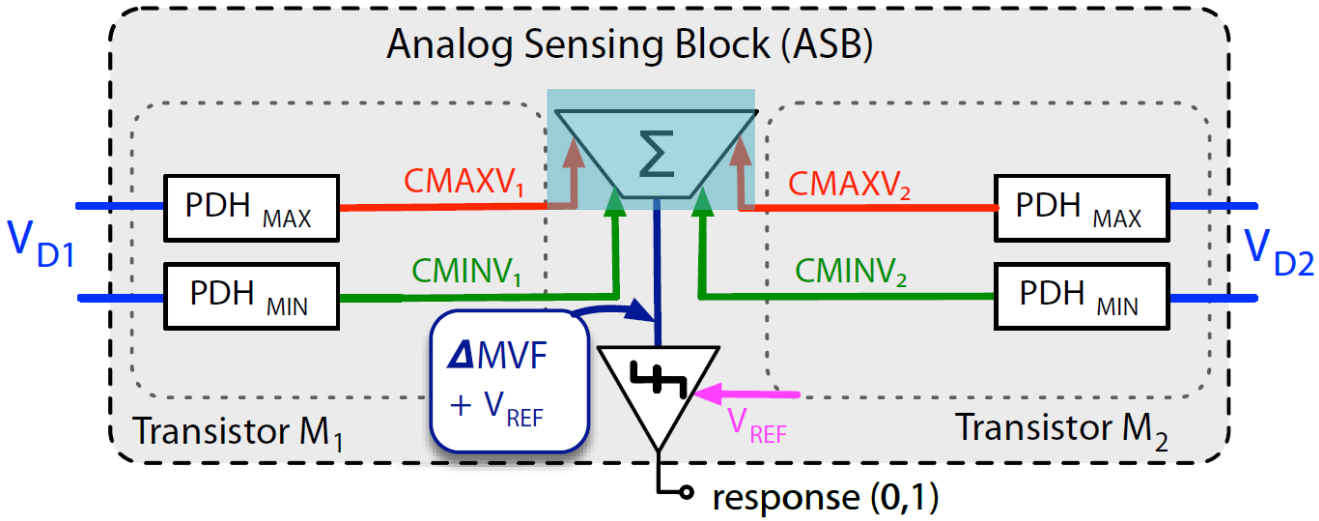


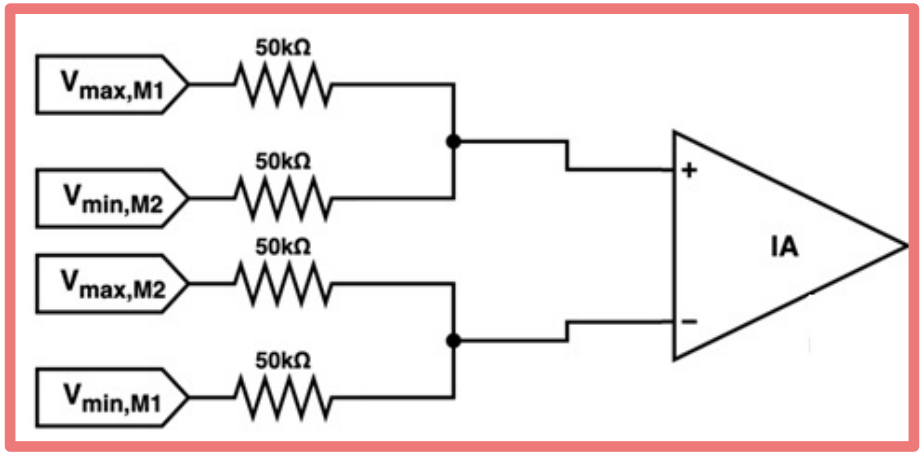
PDH design



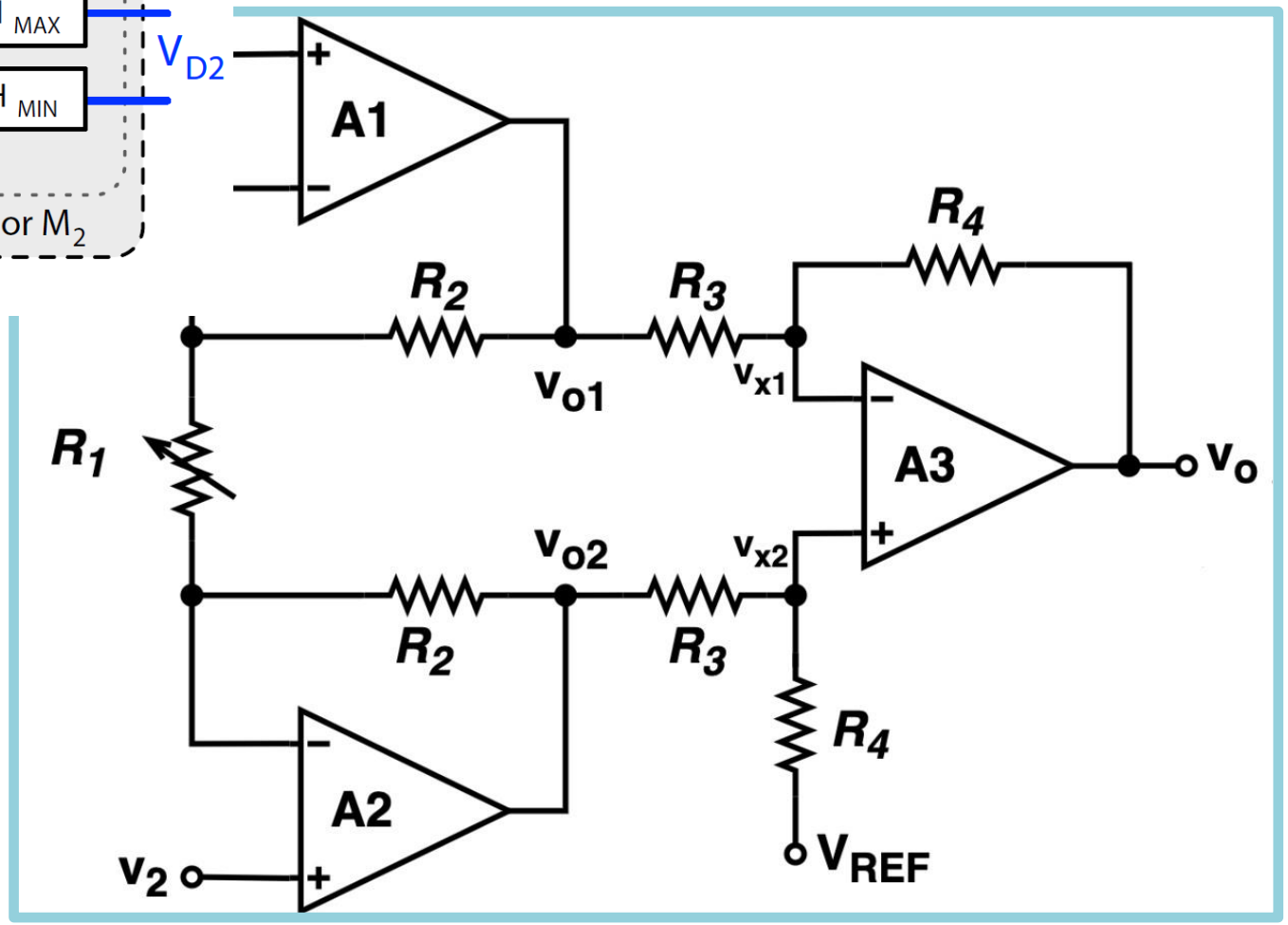
Circuit	Static Error (mV)	
	μ	σ
PDH_{MAX}	0.5319	0.141
PDH_{MIN}	0.8405	0.250

Low-level design: ASB design

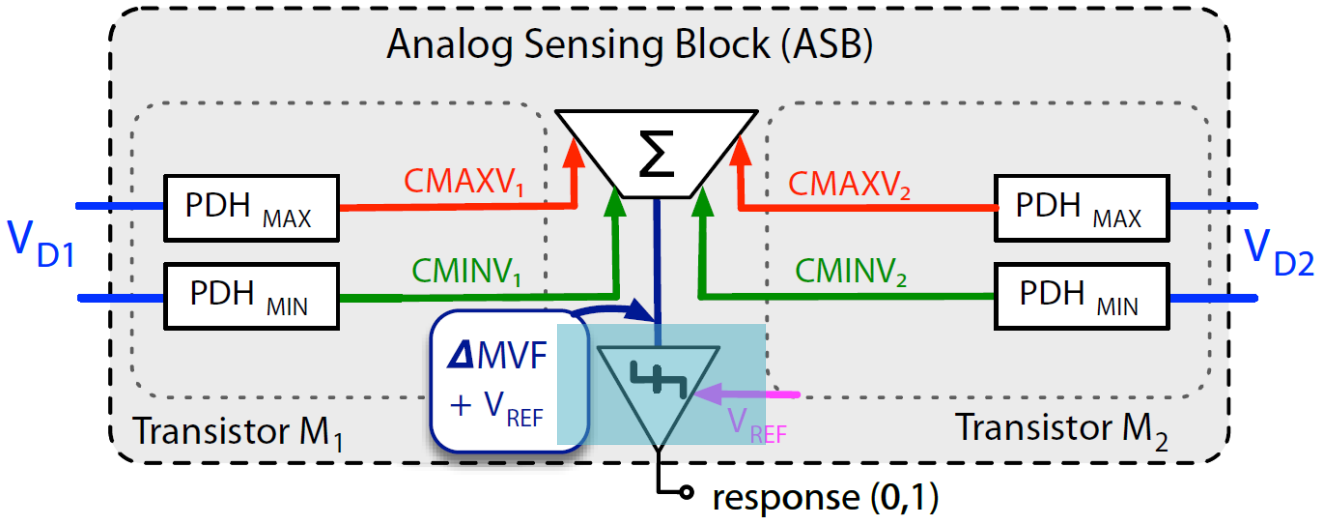




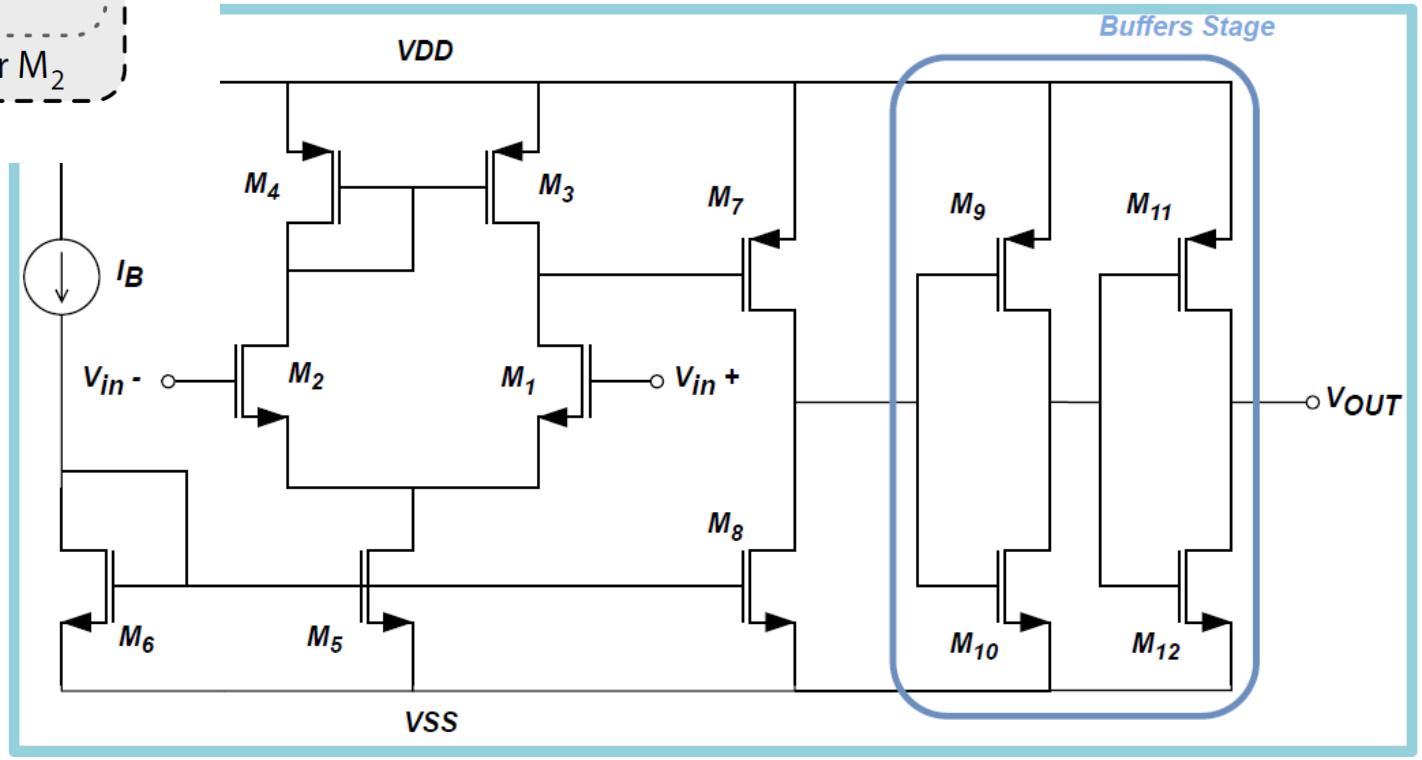
Substractor design



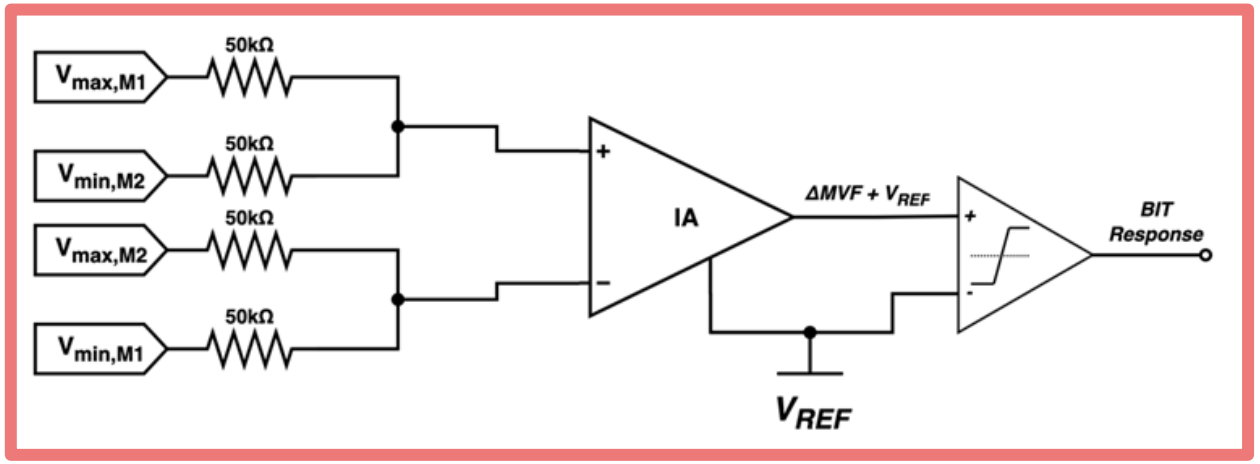
Low-level design: ASB design

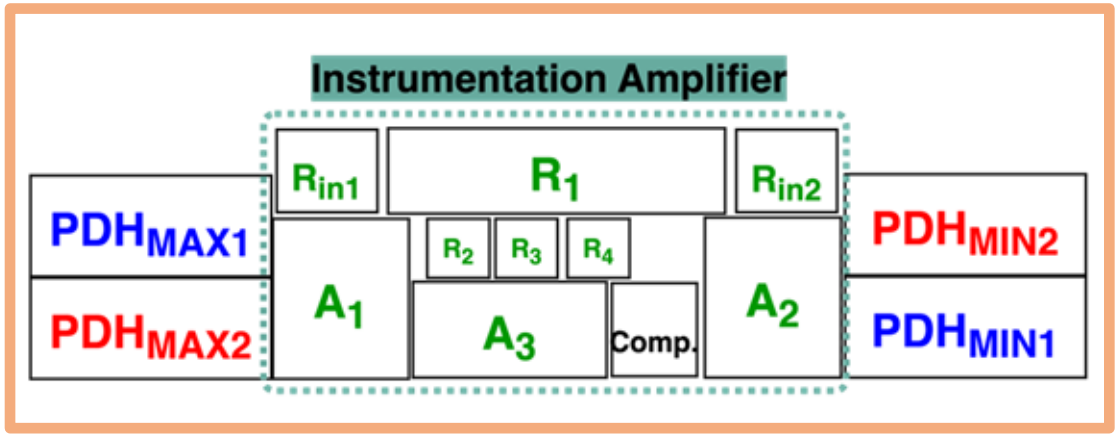


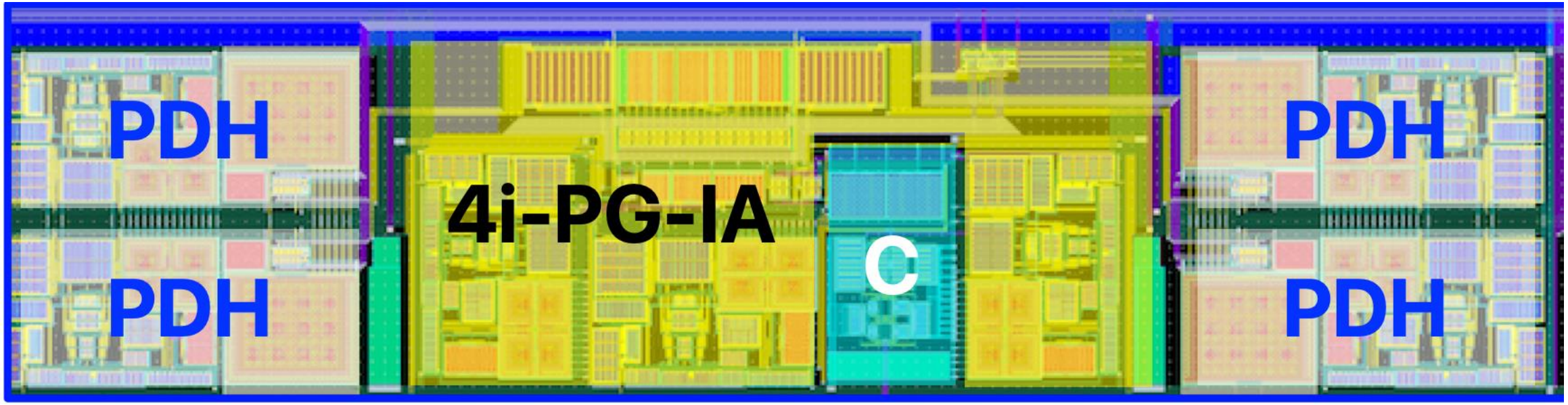
Comparator design



Low-level design: ASB design



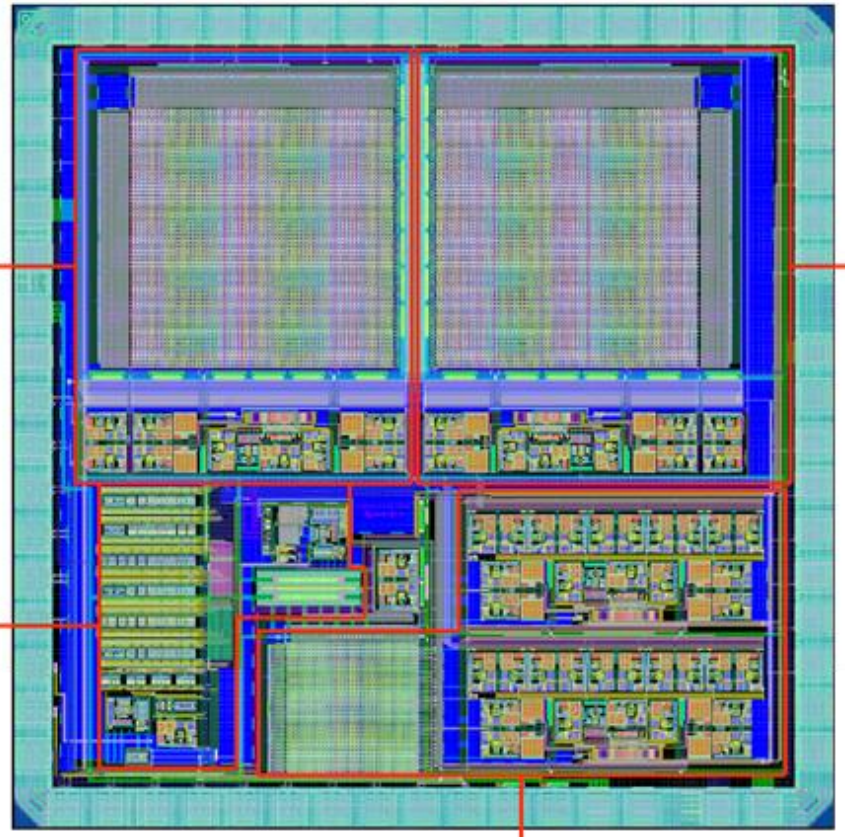




Low-level design: Milestone - I

MILESTONE-I

An RTN-based PUF in UMC 65nm



Primary PUF

A 4096 unit cell PUF, each cell containing:
- CMOS transmission gate
- Transistor (entropy source)
- Digital selection

Secondary PUF

Same as the Primary 4096 PUF

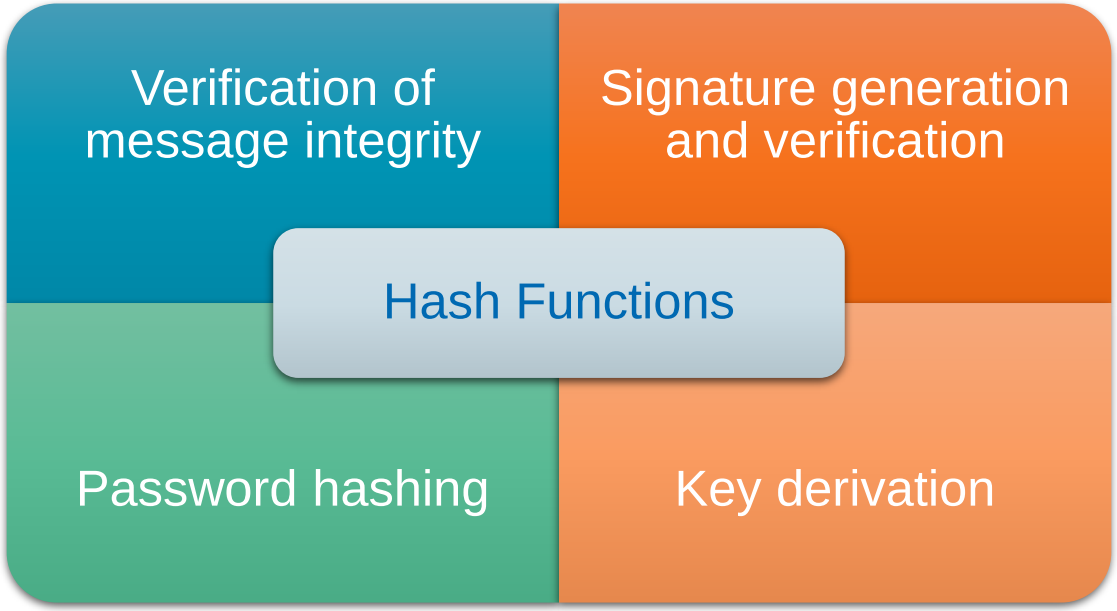
Bias and ICC generation

ICC: current used to bias the Transistors, apply the constant-current method and extract the RTN-based entropy

Test section

Containing:
- A smaller version of the primary PUF (1024 Aray).
- 2 ASB systems
- Accelerated aging capabilities

Hash Functions



- Collision Resistance
- Preimage resistance
- Second Preimage resistance

National Institute of Standards and Technology
U.S. Department of Commerce

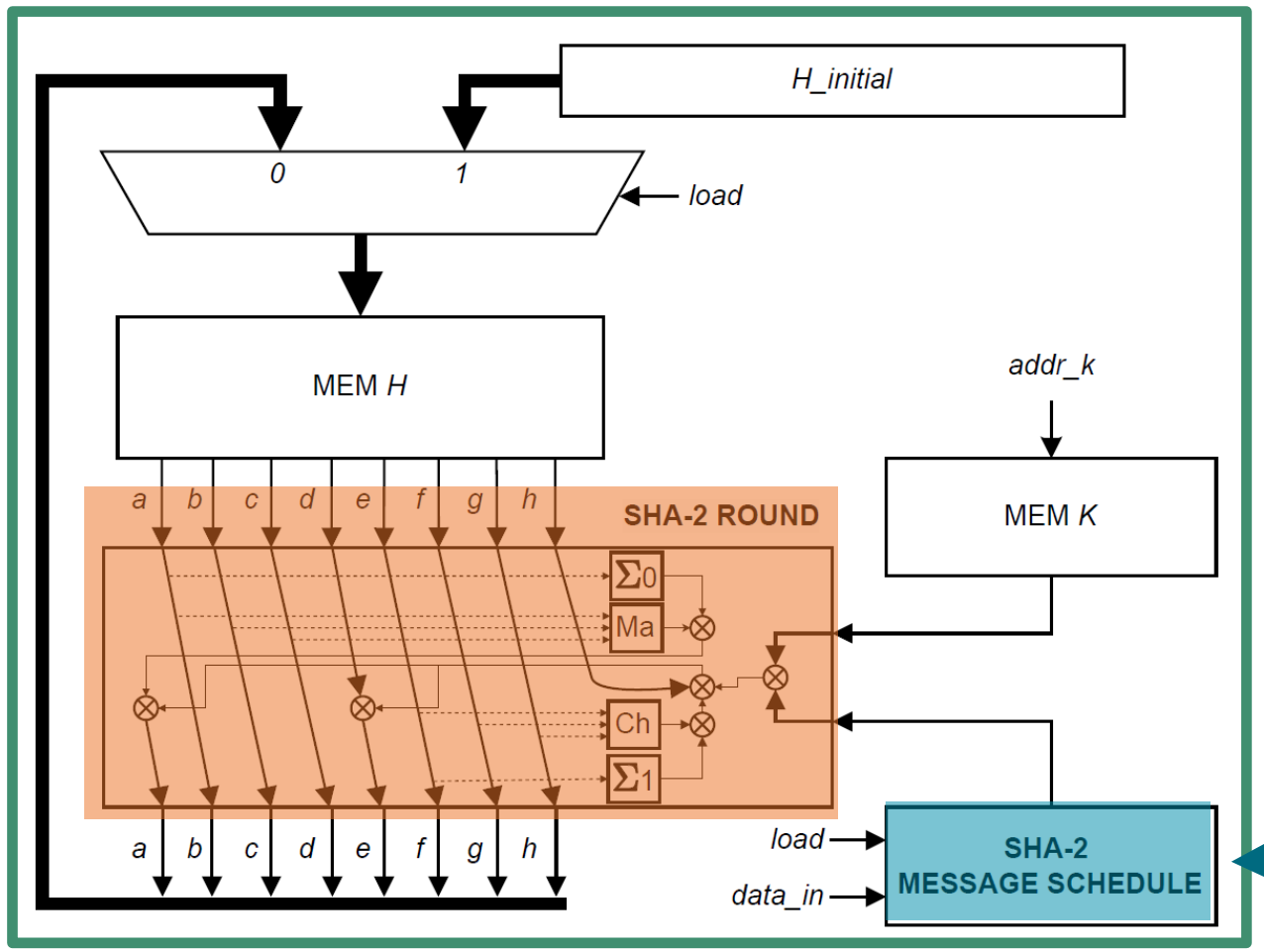
- NIST
  - SHA-2
  - SHA-3

Hash Functions

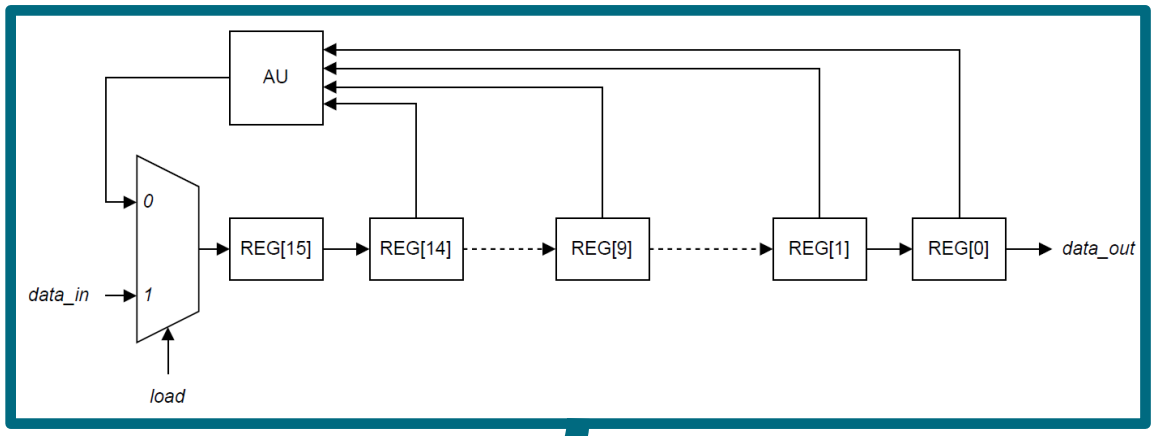
- Competitive trade-off between time and area
- Comparison with the SATO
- Desing of a highly configurable IP: AXI4-Lite
- Drivers to use the IP module
- ASIC implementation of the SHA-256

SHA-2

SHA-2 Core

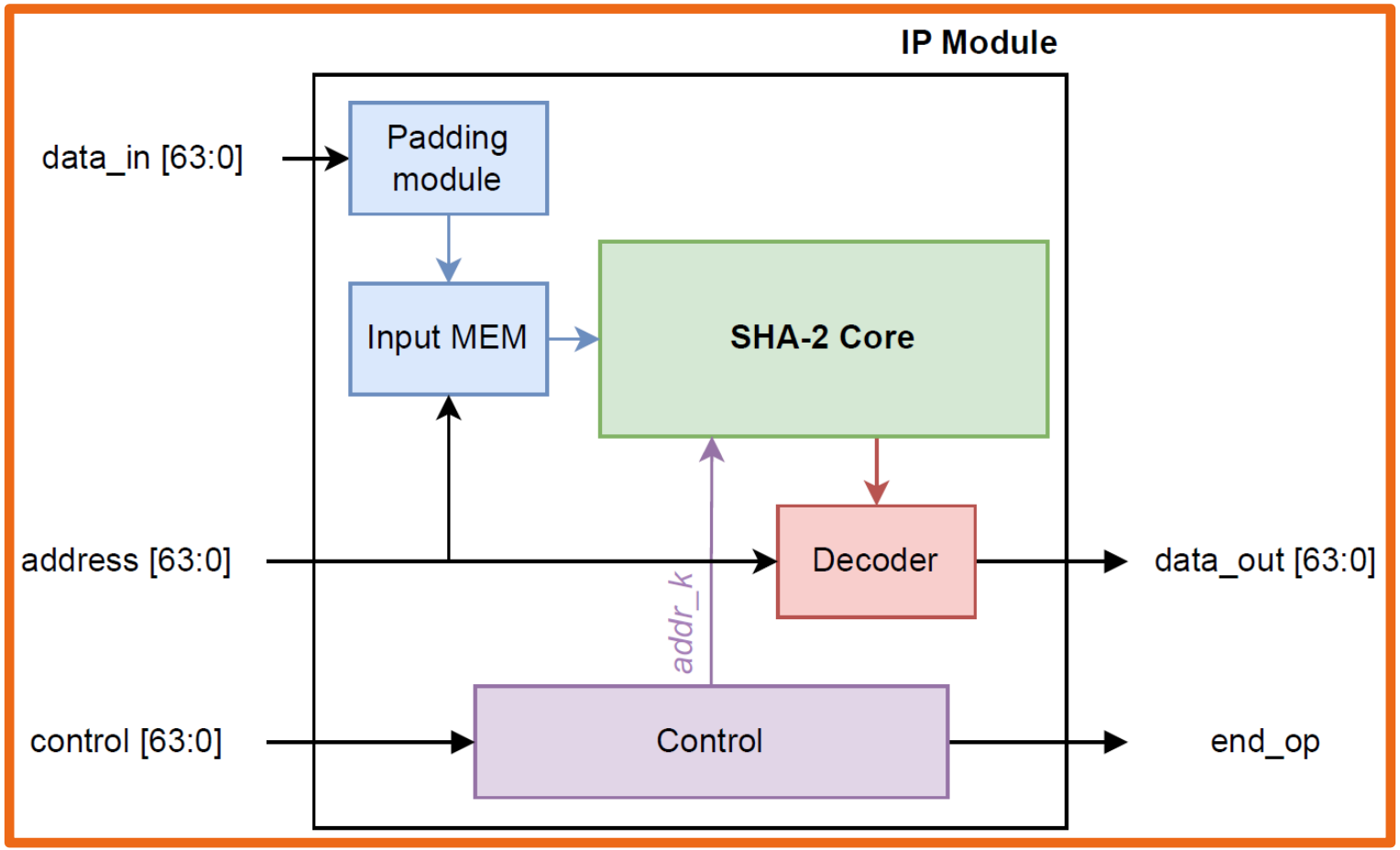


Message Schedule



SHA-2

SHA-2 IP

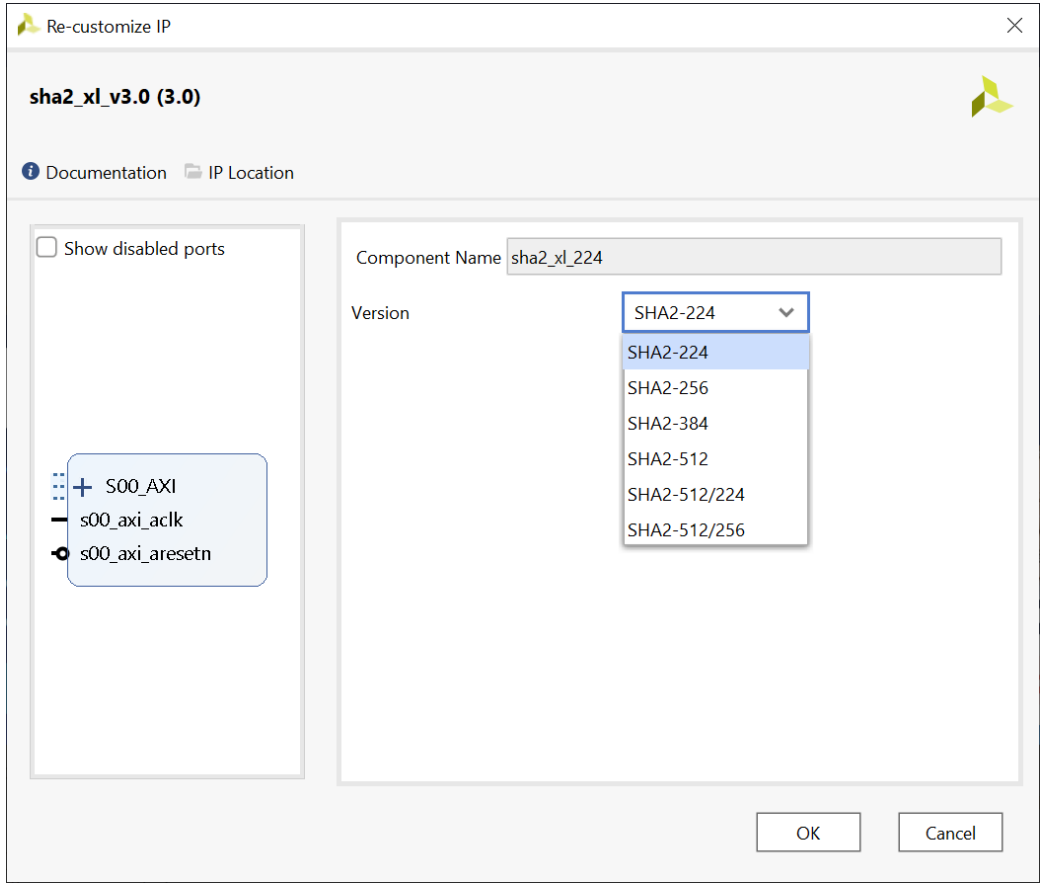

IP Module
data_in [63:0]
Padding module
Input MEM
SHA-2 Core
address [63:0]
Decoder
data_out [63:0]
addr_k
control [63:0]
Control
end_op

SHA-2

Ref	Platform	SHA-2 ver.	Area (Slices)	Frequency (MHz)	Throughput (Gbps)	Efficiency (Mbps/Slice)
[89]	Virtex	SHA-256	755	174.00	1.370	1.830
[90]	Virtex	SHA-256	6136	35.10	2.077	0.338
[91]	Virtex-5	SHA-256	387	202.54	1.580	4.190
[92]	Virtex-4	SHA-256	610	170.75	1.345	2.200
[93]	Virtex-5	SHA-256	1895	411.30	3.290	1.740
This diss.	**Virtex-7**	**SHA-256**	**282**	**191.13**	**1.630**	**5.784**
[89]	Virtex	SHA-512	1667	141.00	1.780	1.010
[91]	Virtex-5	SHA-512	874	176.06	2.200	2.580
[94]	Virtex-5	SHA-512	1080	129.00	0.826	0.765
This diss.	**Virtex-7**	**SHA-512**	**576**	**173.52**	**2.221**	**3.856**

SHA-2

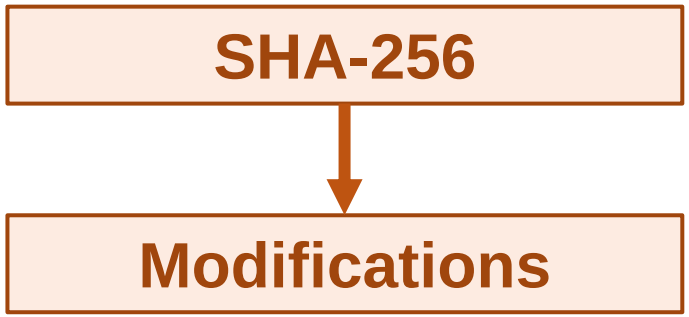
SHA-2 IP Interface



SHA-2 Instance	Message Length < Block Size	Message Length > Block Size
SHA-224	x0.68	x0.87
SHA-256	x0.67	x0.87
SHA-384	x1.54	x2.35
SHA-512	x1.44	x2.32
SHA-512/224	x1.64	x2.35
SHA-512/256	x1.64	x2.37

SHA-256 low-level design

Digital design flow



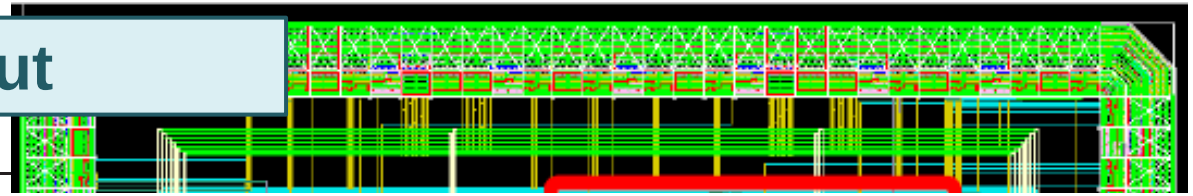
1. Inferring registers and memory cells instead of BRAMs.

2. Adding an interface

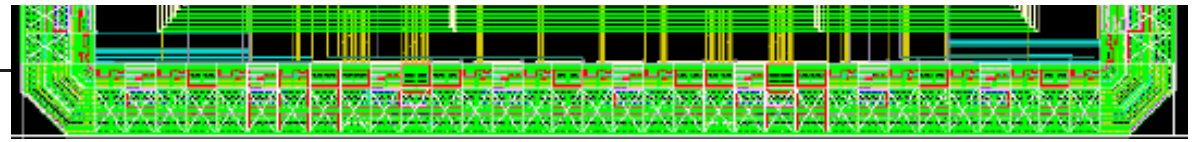
SHA-256 low-level design

Occupation

Layout



Module	Cell Count (GE)	Cell Area(μm^2)	Net Area(μm^2)	Total Area(μm^2)
SIPO	261	1317.240	742.423	***2059.663***
PISO	516	2507.400	1479.523	***3986.923***
Control	52	181.440	2594.252	***2775.692***
SHA-256 (Core)	19350	118787.400	70292.775	***189080.175***
Total	20179	122793.480	75108.973	***197902.453***

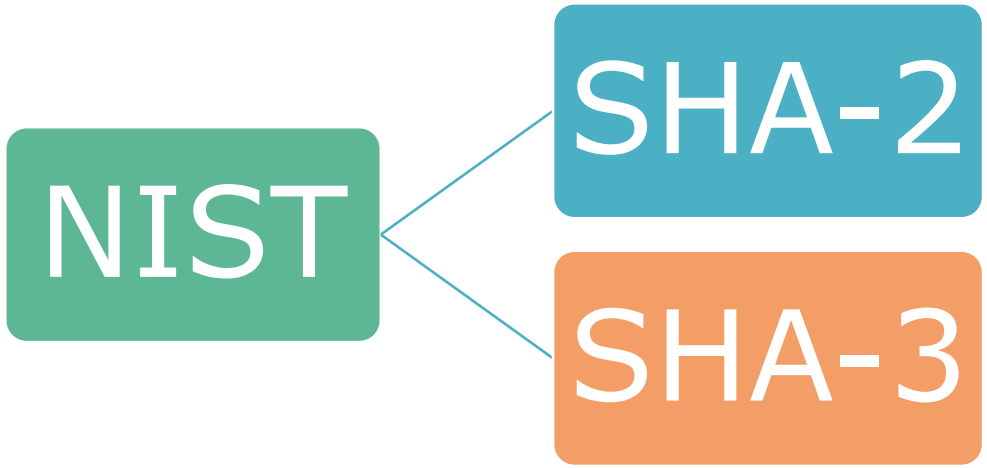


SHA-256 low-level design tests

NIST Test Vector

```
                ############ TEST NIST:          10 ############
LENGTH:                    80
MSG: 74cb9381d89f5aa73368
HASH_NIST: 73d6fad1caaa75b43b21733561fd3958bdc555194a037c2addec19dc2d7a52bd
HASH_SHA2: 73d6fad1caaa75b43b21733561fd3958bdc555194a037c2addec19dc2d7a52bd

CORRECT


                ############ TEST NIST:          69 ############
        LENGTH:                4472
        MSG: 9d64d891d99bb8aba23a29a8f69b32482714e031d31dde3317b046d000f6b7fc421fa8212d91fb66dc46d531
eeafd5ea40302a215351f746c0c42523ba5a3e98bb7b13870d04bf3e0e13425c4fdc11a505ed57c90a90fbc447242b3ee032\
68a29594dd73c705808efc16a059e08dd118b4a34f178175151760de963f89d34c92b12e9b58ace694fadd73a576193b80bf\
ed0074bf5074cfba9e21da980fb366f39e76d1b8073e88ebf2d8d623827bad051f736d02e02688185fbc7ccaea69244fae2c\
15146e63b8ed0cb496f494b4b272bc8aac94c8f0dadb45fd015ab25b210170acd9f05afcc1786b758c6bc87d3d93449497d7\
637a345db161ecc9f00fc9b37677a4de55701f189fba0afba63baaf1584fc36d5819212a5299b39b2c0daad0302aea20d654\
4e3829f0b726b68686e7681ac3a91f543dcb79f2da30aecb30d23e252e7a661fcb619a98056f61d46e1fe473fd3d11b1c6bb\
c80be54d20cee843e0f4f65d7d49032f523e6a4830abacf56de9f46bd7c86865ad4359230a9f5dafc928b61c9456a1fbf142\
7a53cb82dff264eb2de7f9feaf739a47aa64c4a2fd70772f026a33cf1451e852a9e47ae083a159f62e23c0cae8402f775d84\
f77044204b765fb8e418d6cbb7dd7dacc74b148cbda95991f4c3cf65dd60e6f61b8dce59e6ad127b2dda65b3d0416a0f4939\
2f1f107354c4de6fa14f1482db5a9961f867b921ef33697a4db4d22cf37e69211fd2f2c2944f16252a86755baf0509835ee4\
33733a743f8f0b493e0eae8cb
        HASH_NIST: db593a375cb27df689cd78b5154949e5bc30094a05d704c0295d547385176662
        HASH_SHA2: db593a375cb27df689cd78b5154949e5bc30094a05d704c0295d547385176662

        CORRECT
```

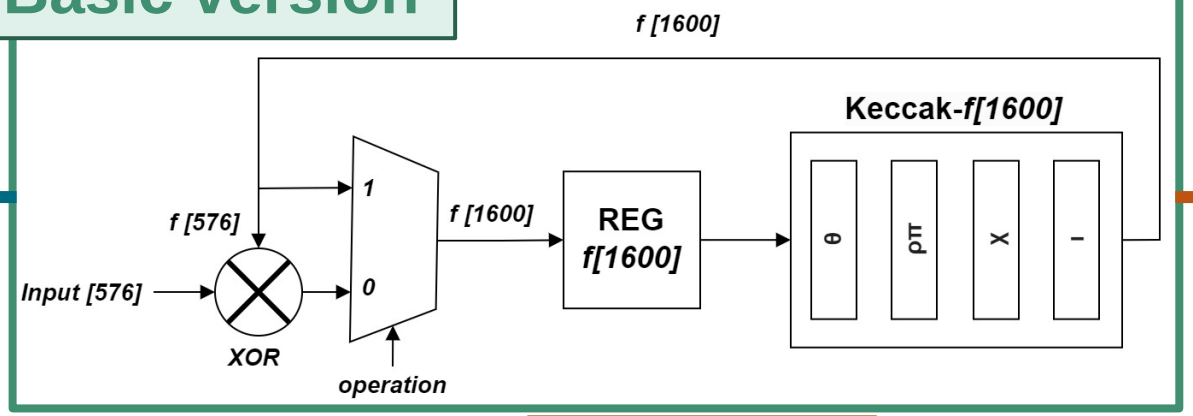
SHA-3



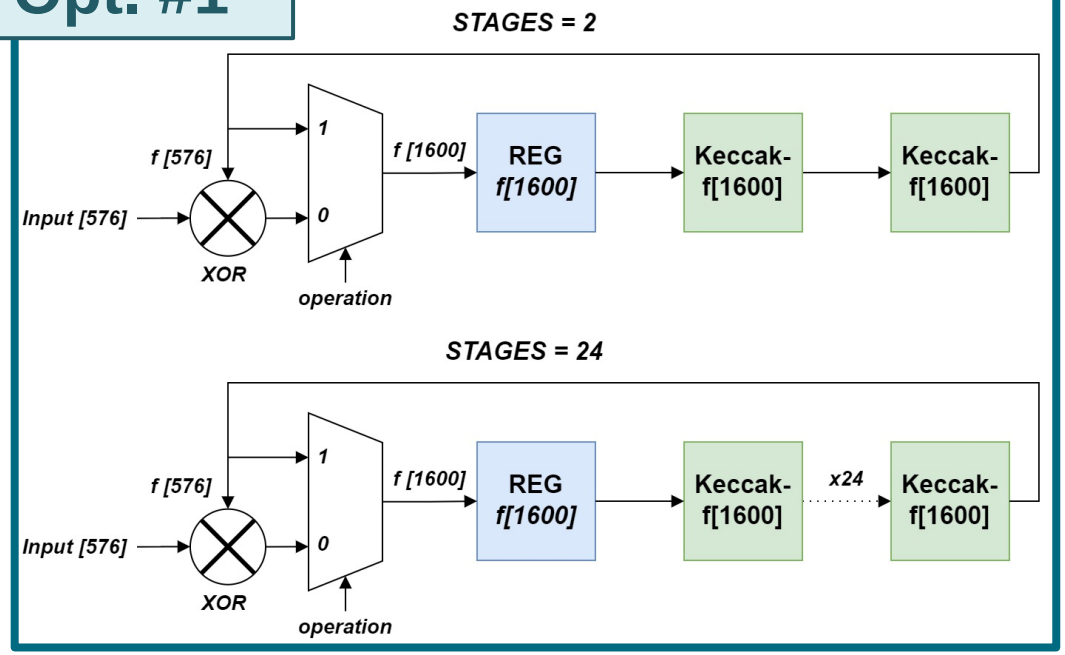
- SHA-3 family has relatively fewer implementation costs and is much faster
- SHA-3 family is noted as PQC primitive
- SHA-3 is based in the Keccak algorithm

SHA-3

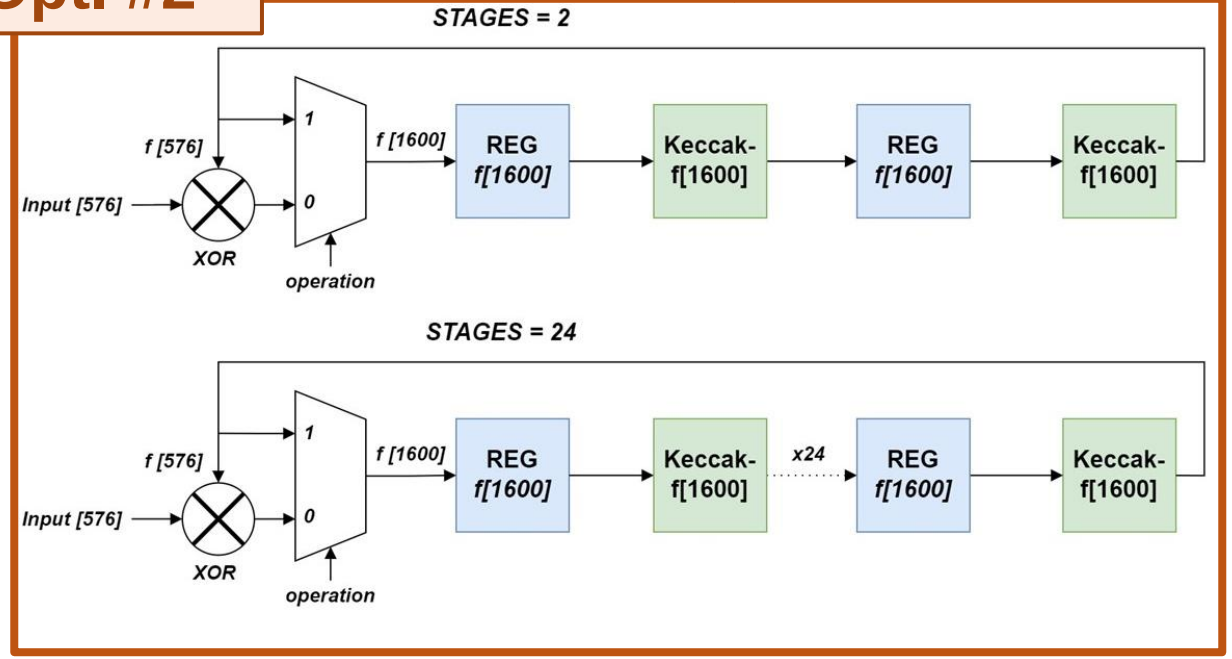
Basic version



Opt. #1

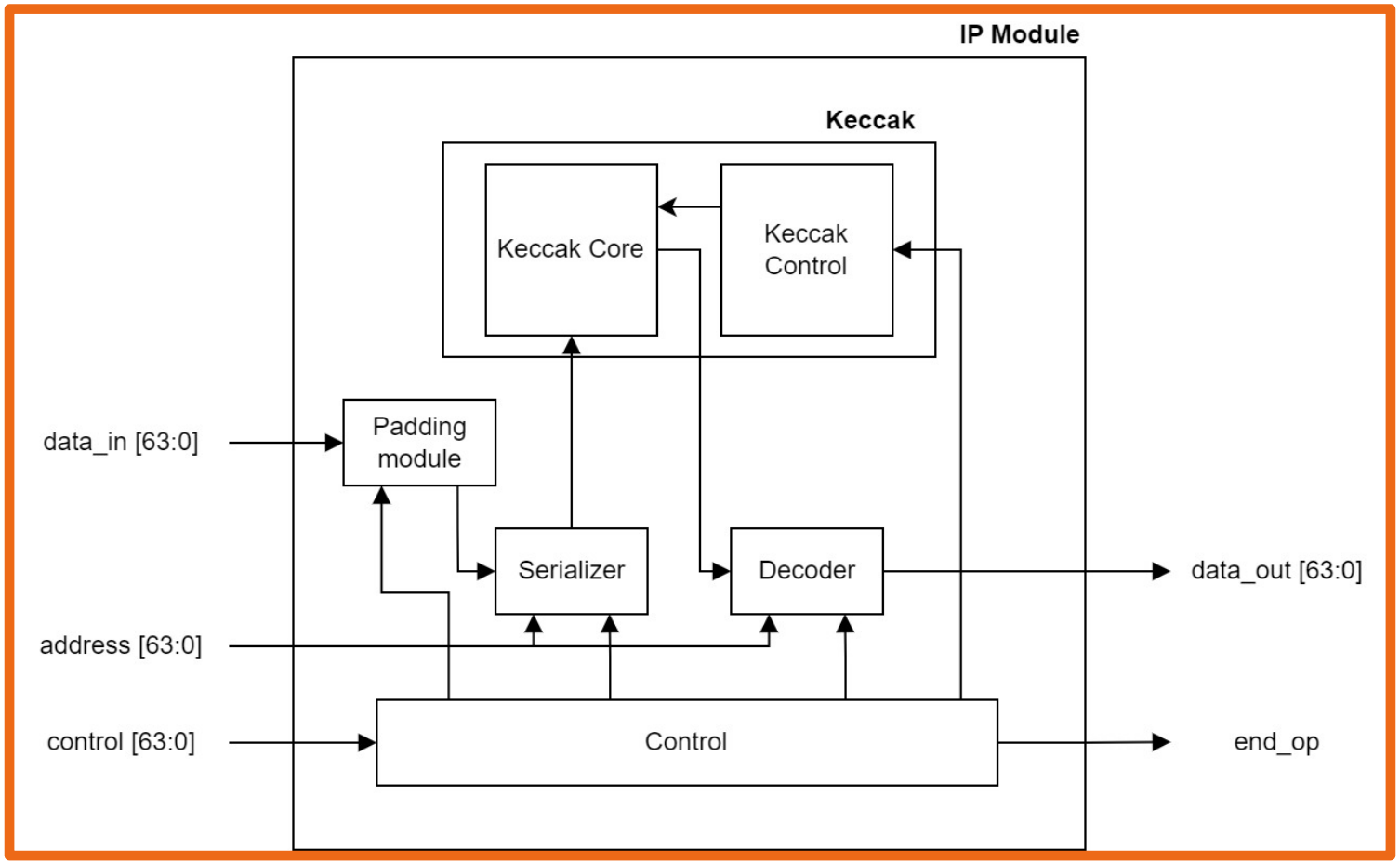


Opt. #2



SHA-3

SHA-3 IP

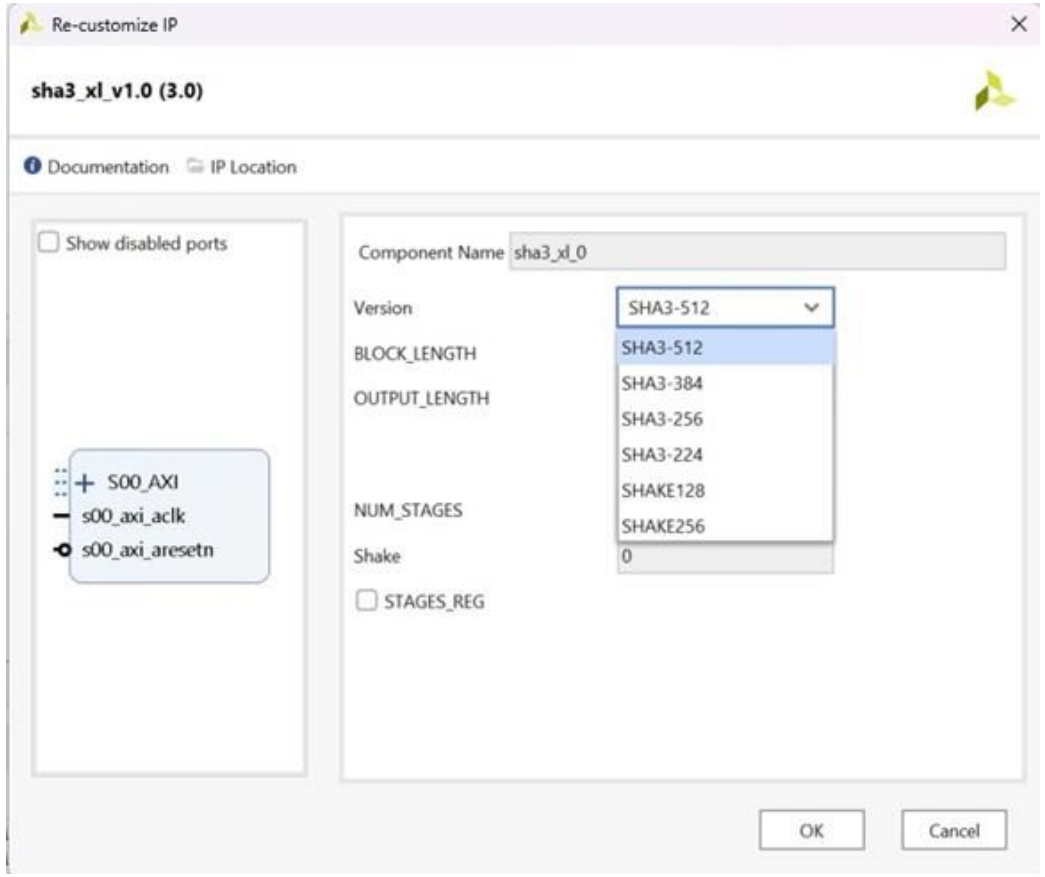

IP Module
Keccak
Keccak Core
Keccak Control
data_in [63:0]
Padding module
Serializer
Decoder
data_out [63:0]
address [63:0]
control [63:0]
Control
end_op

SHA-3

Ref	*Platform*	*SHA-3 ver.*	*Area (Slices)*	*Throughput (Gbps)*	*Efficiency (Mbps/Slice)*
[88]	Virtex-6	SHA3-512	1406	16.51	11.47
[107]	Virtex-6	SHA3-512	2296	18.78	8.17
[110]	Virtex-5	SHA3-512	1163	7.80	6.06
This diss.	**Virtex-7**	**SHA3-512**	**1456**	**17.97**	**12.34**
[109]	Virtex-7	SHA3-256	1618	20.08	12.90
[108]	Virtex-7	SHA3-256	1418	14.30	11.97
This diss.	**Virtex-7**	**SHA3-256**	**1566**	**29.25**	**18.68**
[111]	Zynq-7000	SHA3-224	1424	12.8	8.98
This diss.	**Zynq-7000**	**SHA3-224**	**1577**	**30.00**	**19.02**

SHA-3

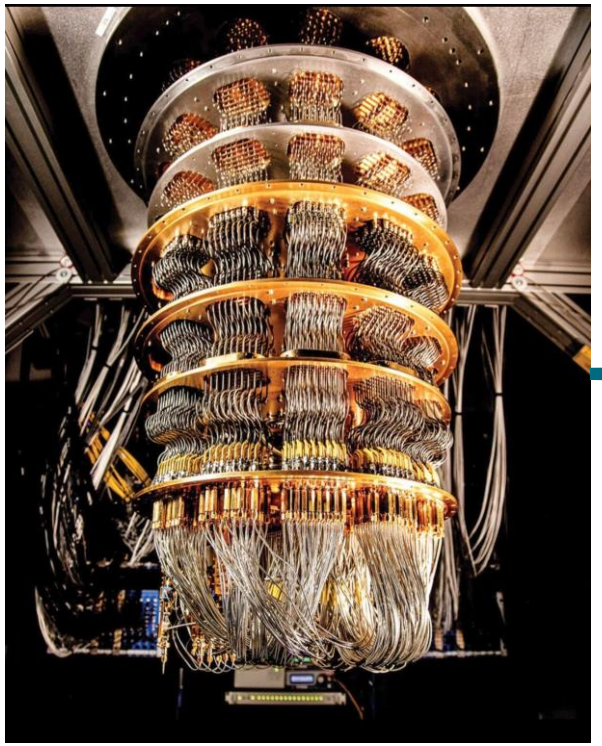
SHA-3 IP Interface



SHA-3 Instance	Message Length < Block Size	Message Length > Block Size
SHA3-224	x6.23	x10.89
SHA3-256	x6.41	x11.71
SHA3-384	x6.77	x14.54
SHA3-512	x7.33	x19.93
SHAKE128	x2.80	x8.18
SHAKE256	x3.07	x9.64

Post-Quantum Cryptography

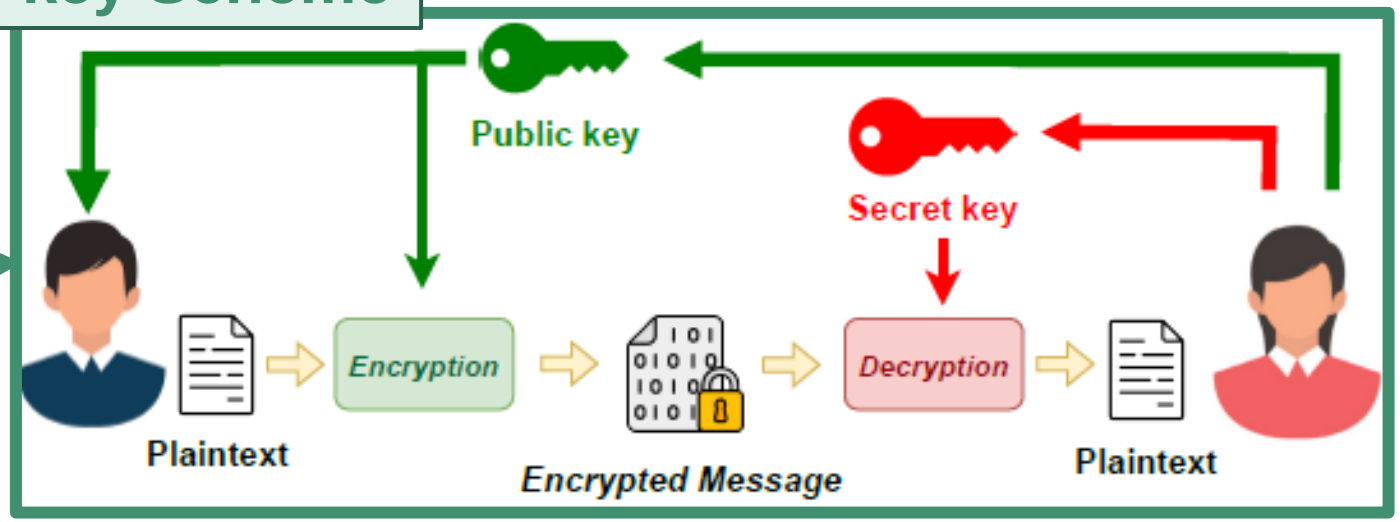
Quantum Computers



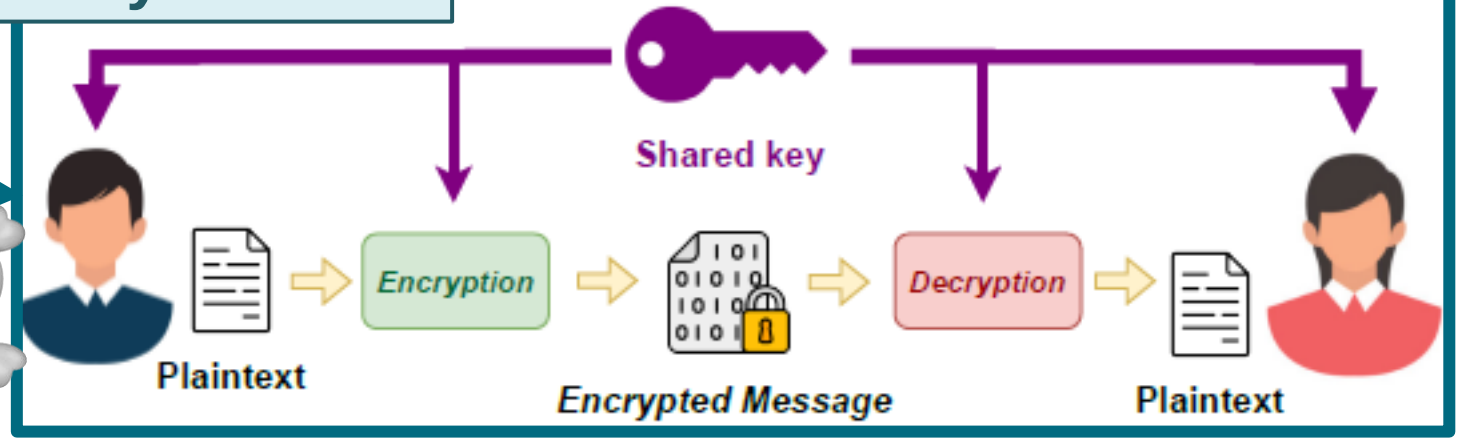
Shor Alg.

Grove Alg.

Public key Scheme



Secret key Scheme



Post-Quantum Cryptography

Shor Alg.

Grove Alg.

Post-Quantum Cryptography

NIST National Institute of Standards and Technology U.S. Department of Commerce

Third Round (2020) → Selected Algorithms (2022) → First Draft (2023)

→ NTRU

OpenSSL Cryptography and SSL/TLS Toolkit

Google Chrome

Implementing the polynomial multiplier in HW

PQC: NTRU

NTRU (IEEE 1363.1) → **Accelerating N cycles when a 0 is found in the blind polynomial**

Can we use this strategy with NTRU Round-3 ?

The number of non-zeros of the blind polynomial is not constant

Disclosing information that has to be hided through **timing attacks**

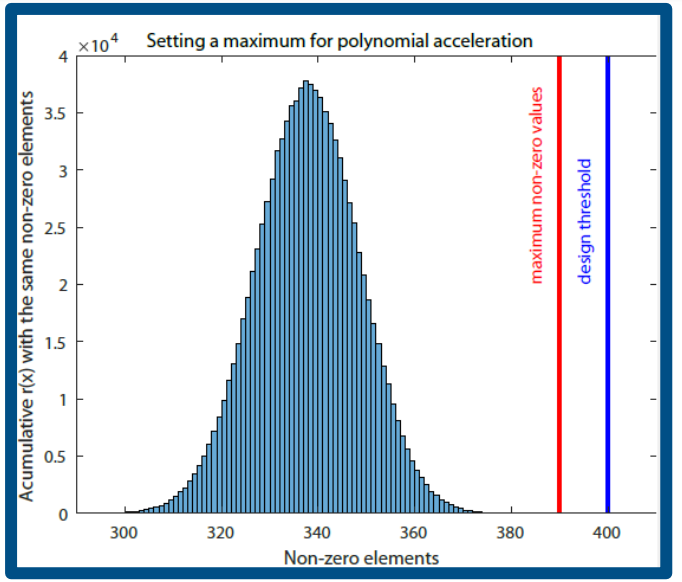
NTRU (Round-3)

↓

New Strategy

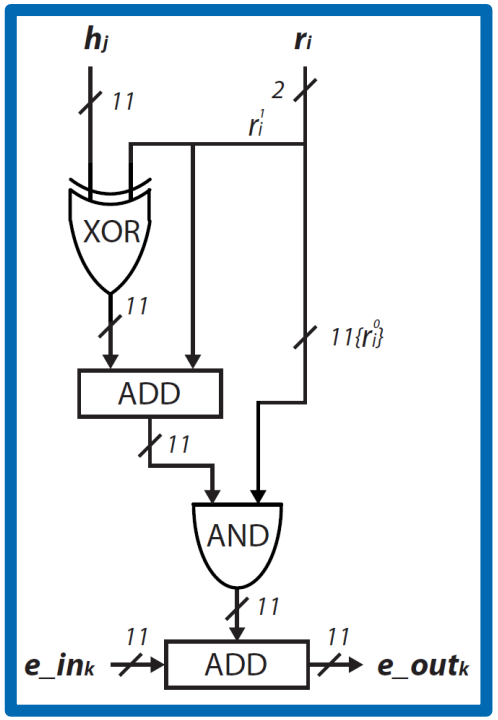
↓

Confident Limit



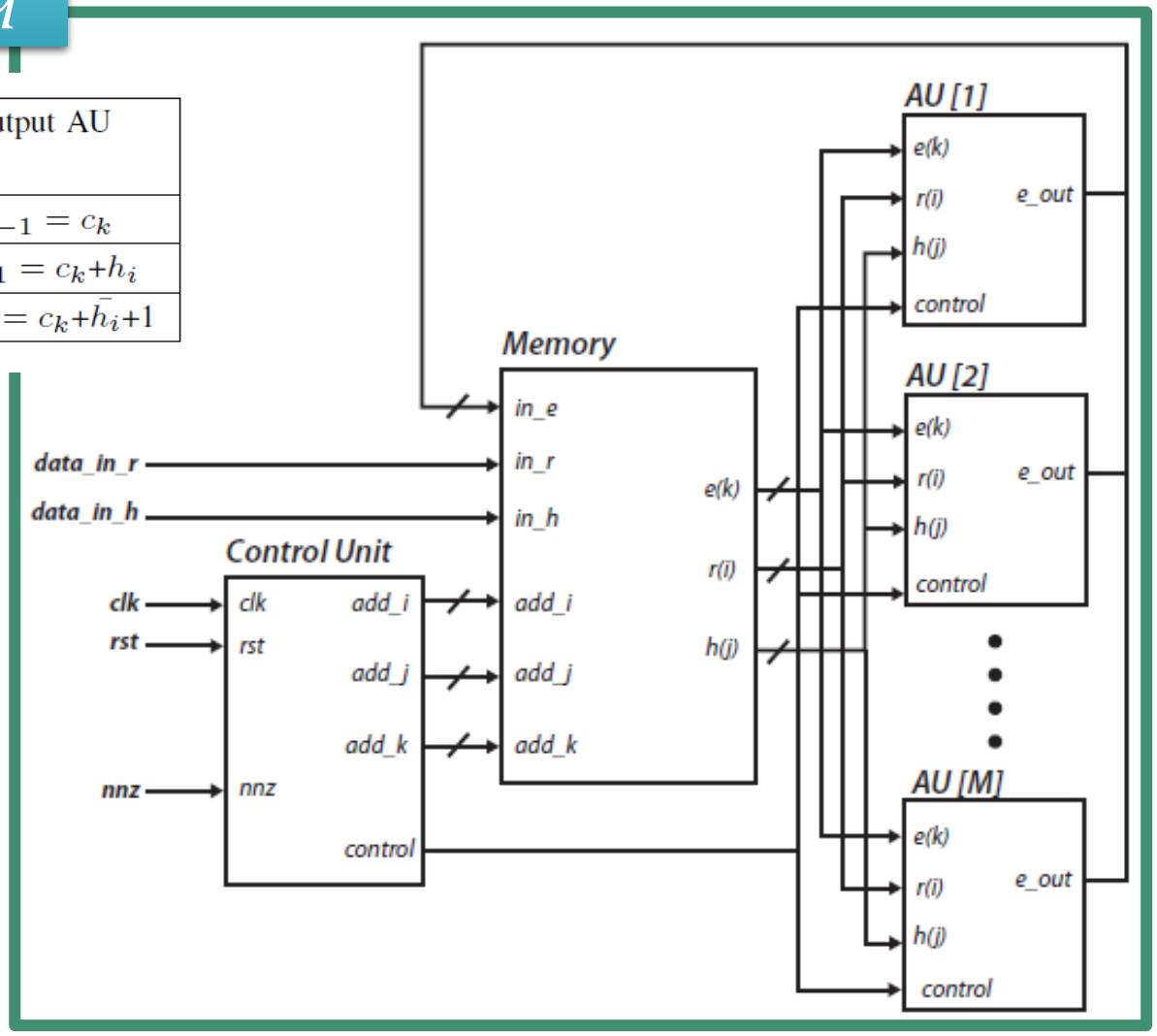
PQC: NTRU

$$e(x) = \left(h(x) \times r(x) + m(x)\right) mod\ q$$



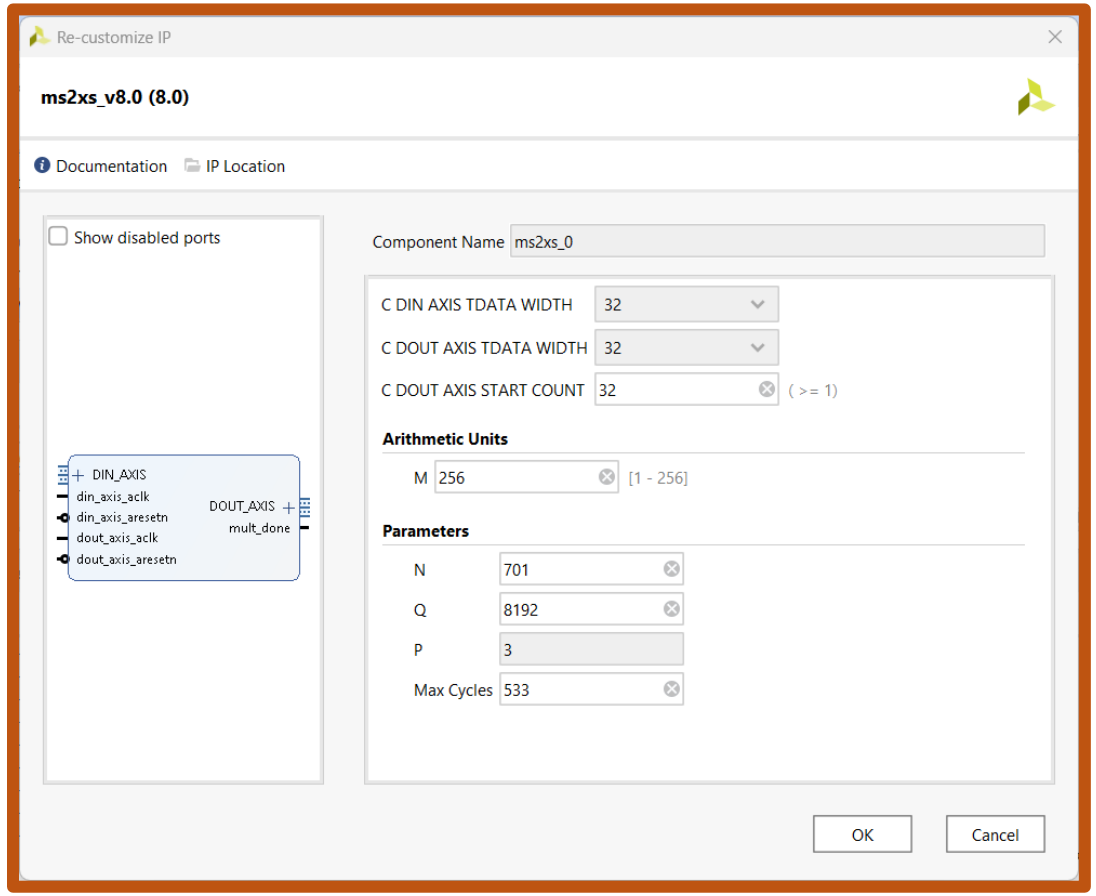
Coefficient value r_j	Control Signal $[r_j^1 r_j^0]$	Output AU
0	00	$c_{k-1} = c_k$
1	01	$c_{k-1} = c_k + h_i$
-1	11	$c_{k-1} = c_k + \bar{h}_i + 1$

Parallelization strategy



PQC: NTRU

IP Interface



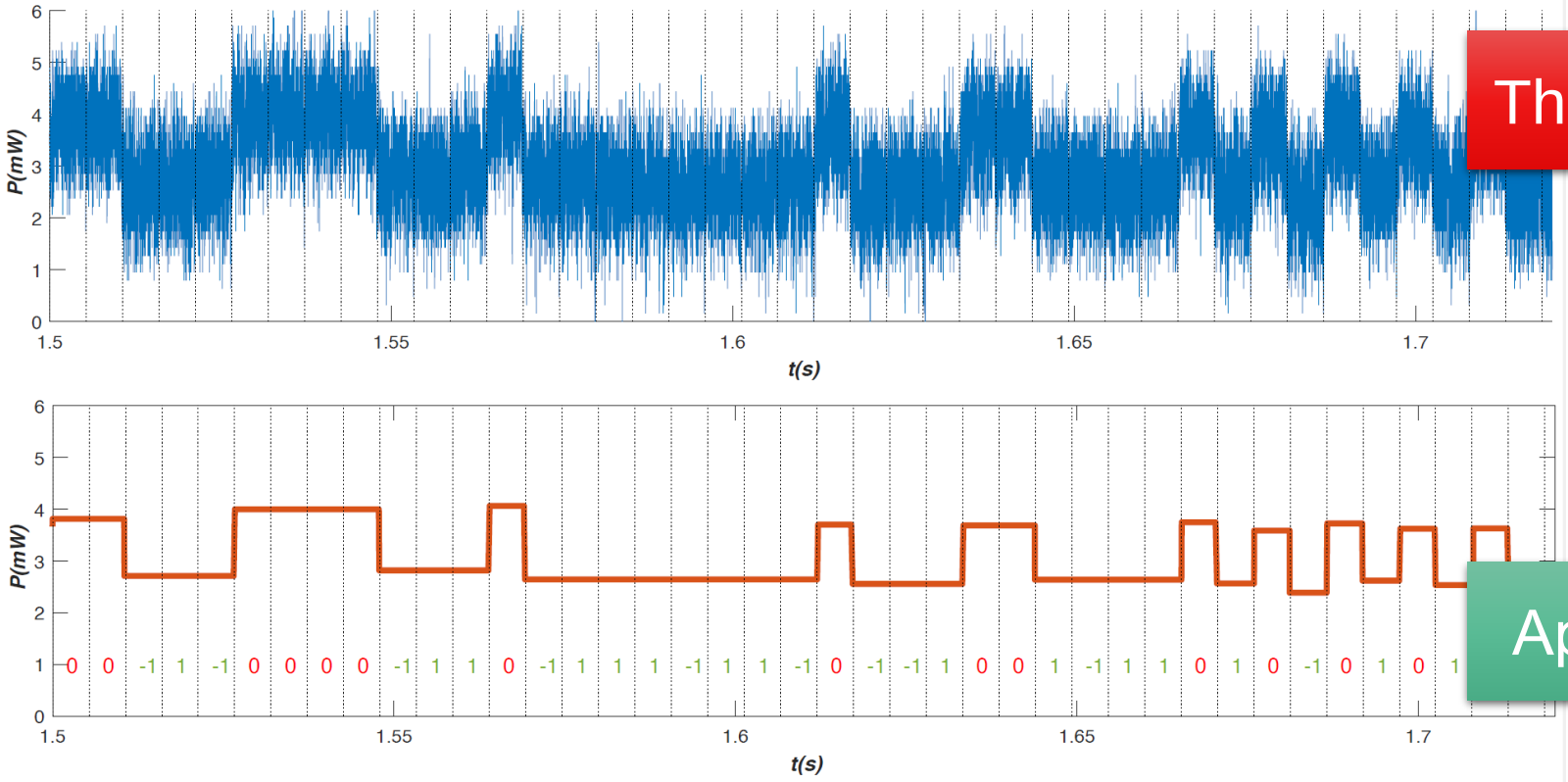
N	max_{coef}	Max. Eff.	M	LUTs	FFs	BRAM	Acc. (x)
509	400	LUT	4	285	119	4.5	20.67
		FF	7	437	124	7.5	30.28
		BRAM	8	584	205	4.5	32.81
	509	LUT	6	348	105	6.5	23.40
		FF	7	399	106	7.5	26.00
		BRAM	8	556	187	4.5	28.35
677	516	LUT	8	476	128	8.5	37.46
		FF	10	582	133	10.5	43.08
		BRAM	11	801	251	6.0	45.49
	677	LUT	8	451	108	8.5	31.23
		FF	10	558	113	10.5	36.40
		BRAM	11	773	231	6.0	38.64
821	625	LUT	8	510	129	8.5	40.06
		FF	12	713	133	12.5	51.89
		BRAM	13	981	287	7.0	54.25
	821	LUT	8	481	109	8.5	32.94
		FF	12	679	113	12.5	43.74
		BRAM	13	929	267	7.0	45.99
701	533	LUT	6	433	130	6.5	32.26
		FF	10	634	133	10.5	43.54
		BRAM	11	881	273	6.0	46.32
	701	LUT	6	403	110	6.5	25.52
		FF	10	605	113	10.5	36.62
		BRAM	11	854	253	6.0	39.18

PQC: NTRU SCA

Coefficient value r_j	Control Signal $[r_j^1 r_j^0]$	Output AU
0	00	$c_{k-1} = c_k$
1	01	$c_{k-1} = c_k + h_i$
-1	11	$c_{k-1} = c_k + \bar{h}_i + 1$

Can that lead somehow to a leak of information ?

Single Power Analysis (SPA)

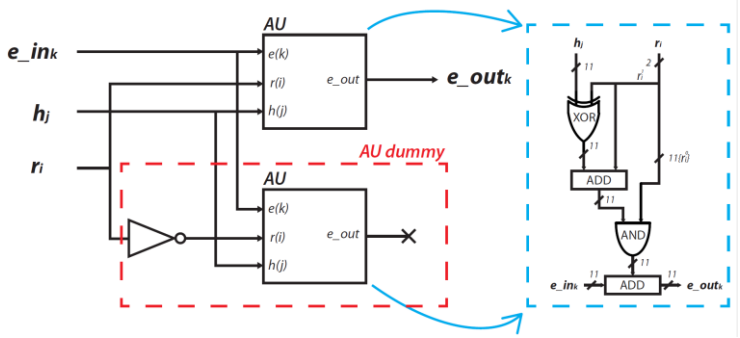


There is leakage of information

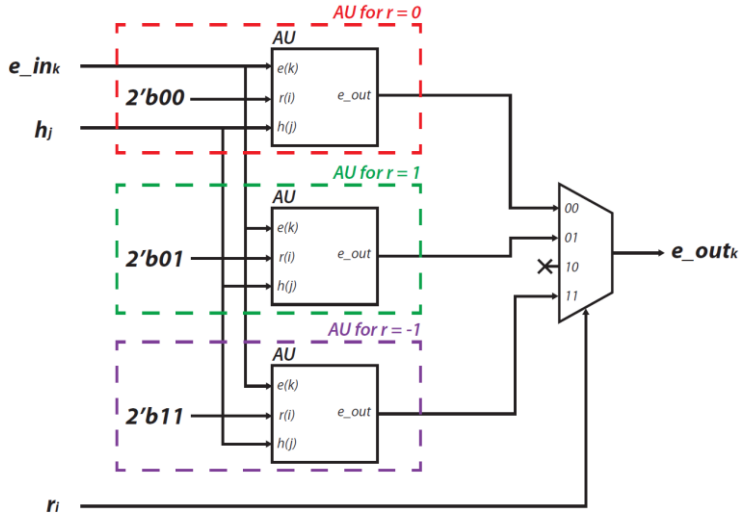
Applying countermeasures …

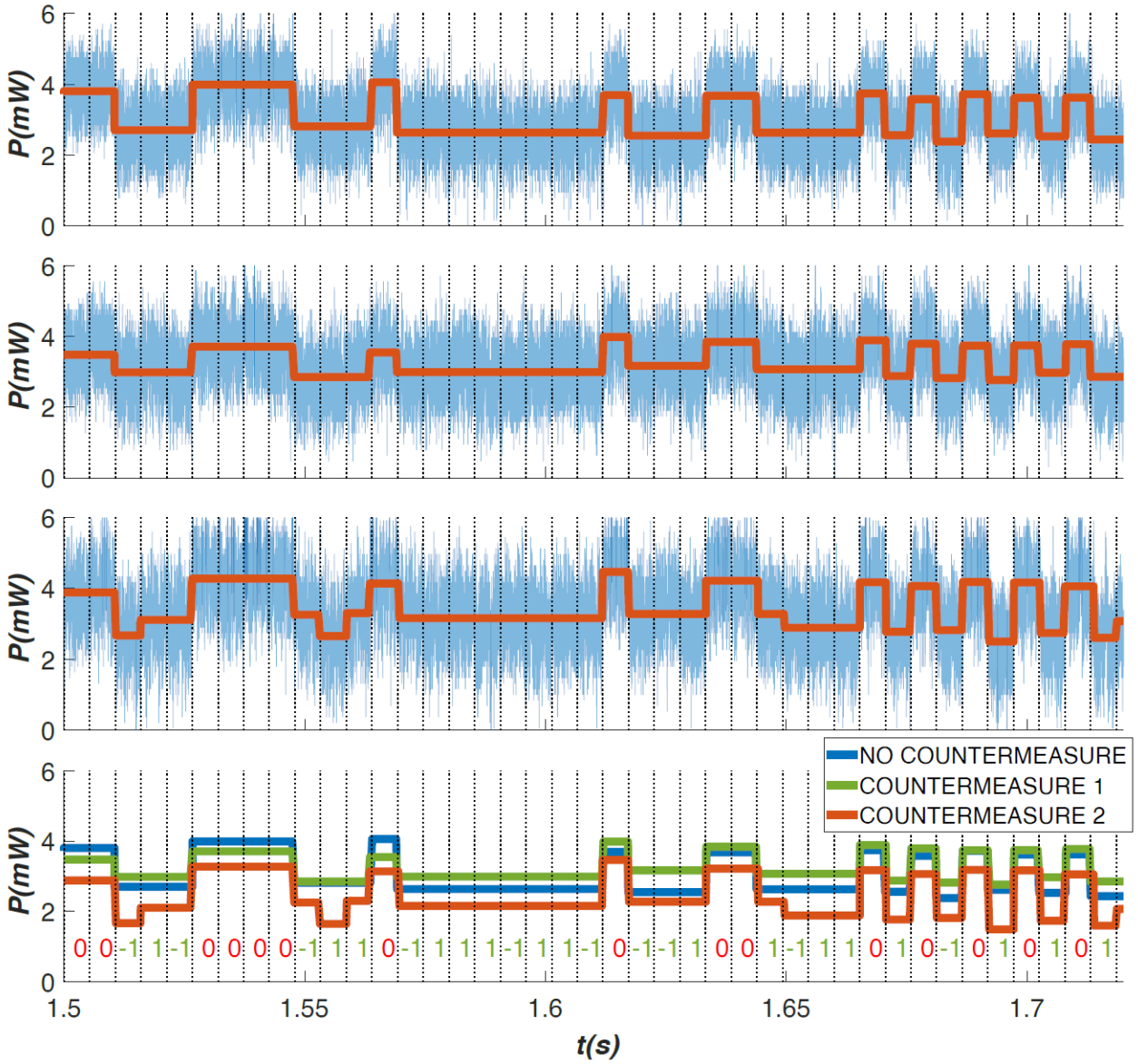
PQC: NTRU SCA

First Countermeasure



Second Countermeasure



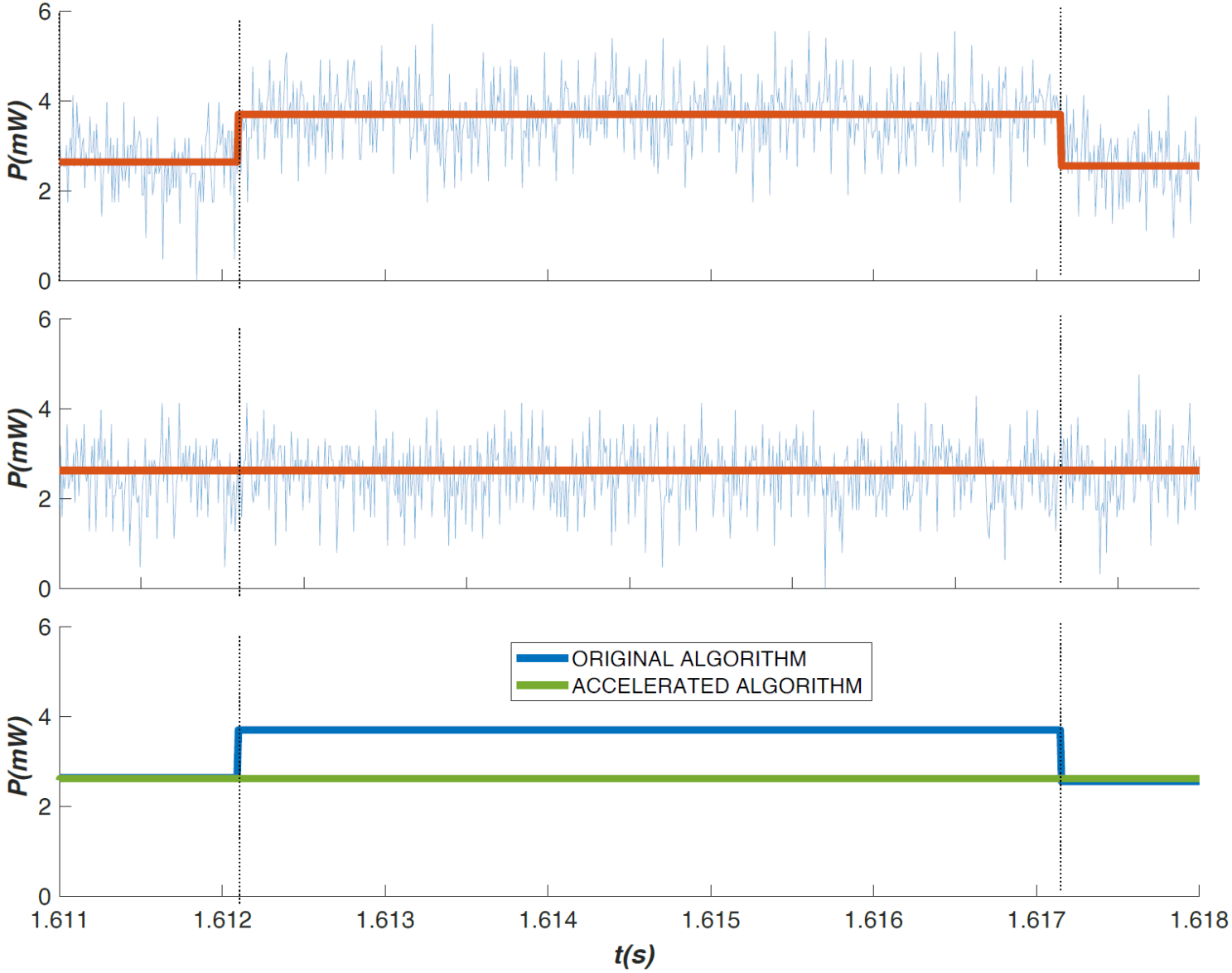


PQC: NTRU SCA

Third Countermeasure

Not using 0's in the multiplication operation

More closer ...



Proof-of-Concept

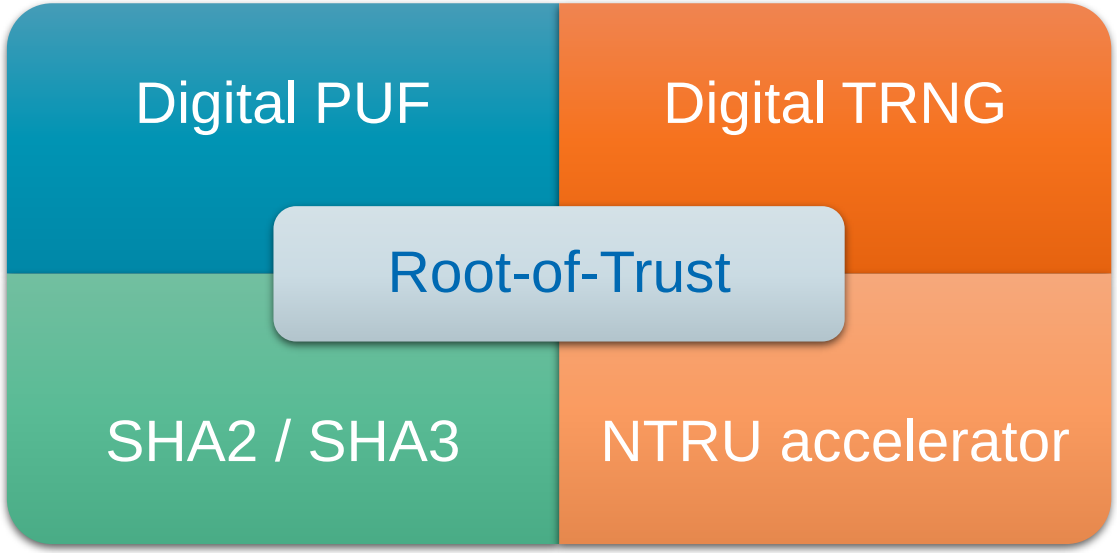
Combine modules in a RoT

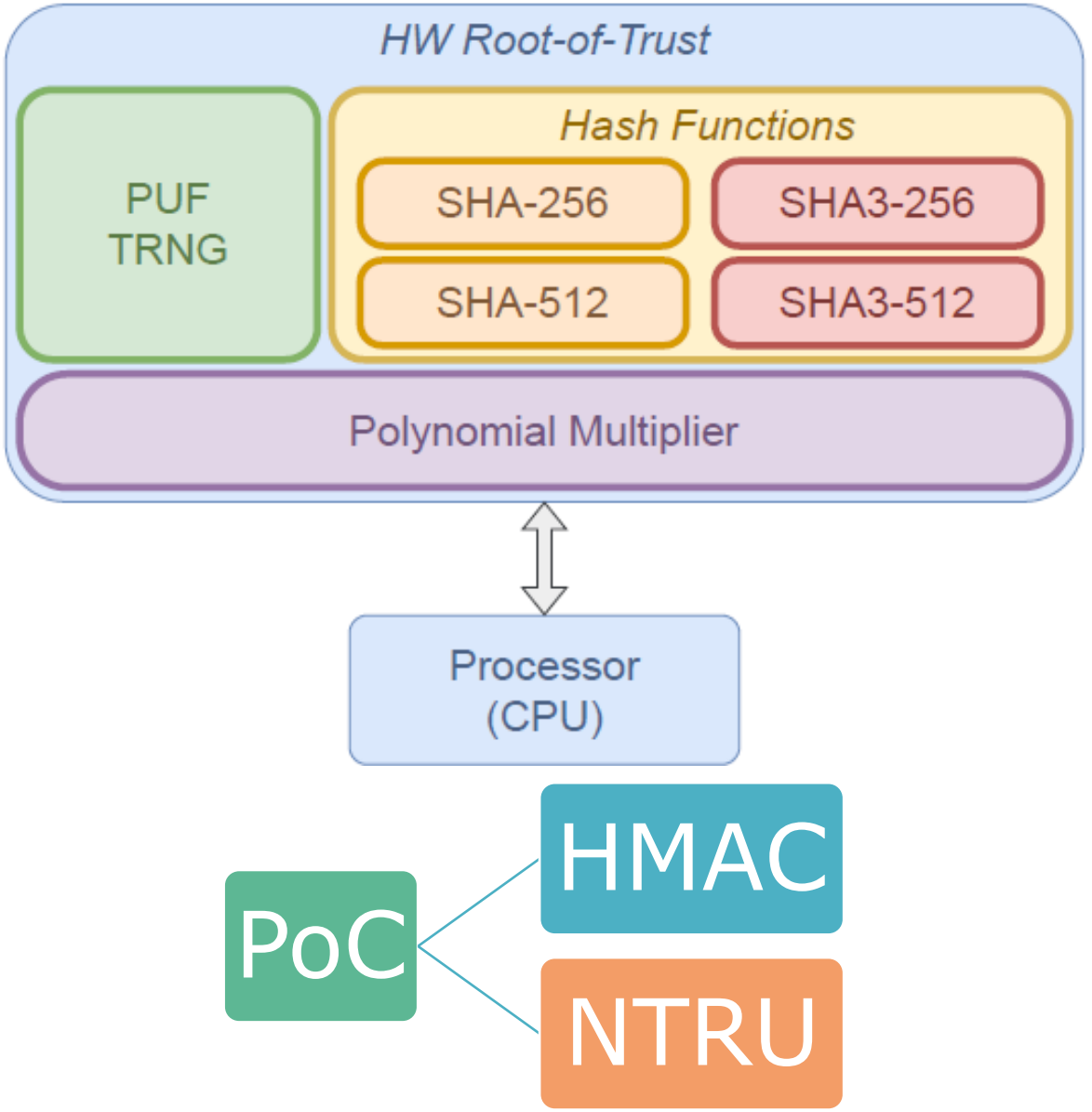
↓

System-on-Chip (SoC)

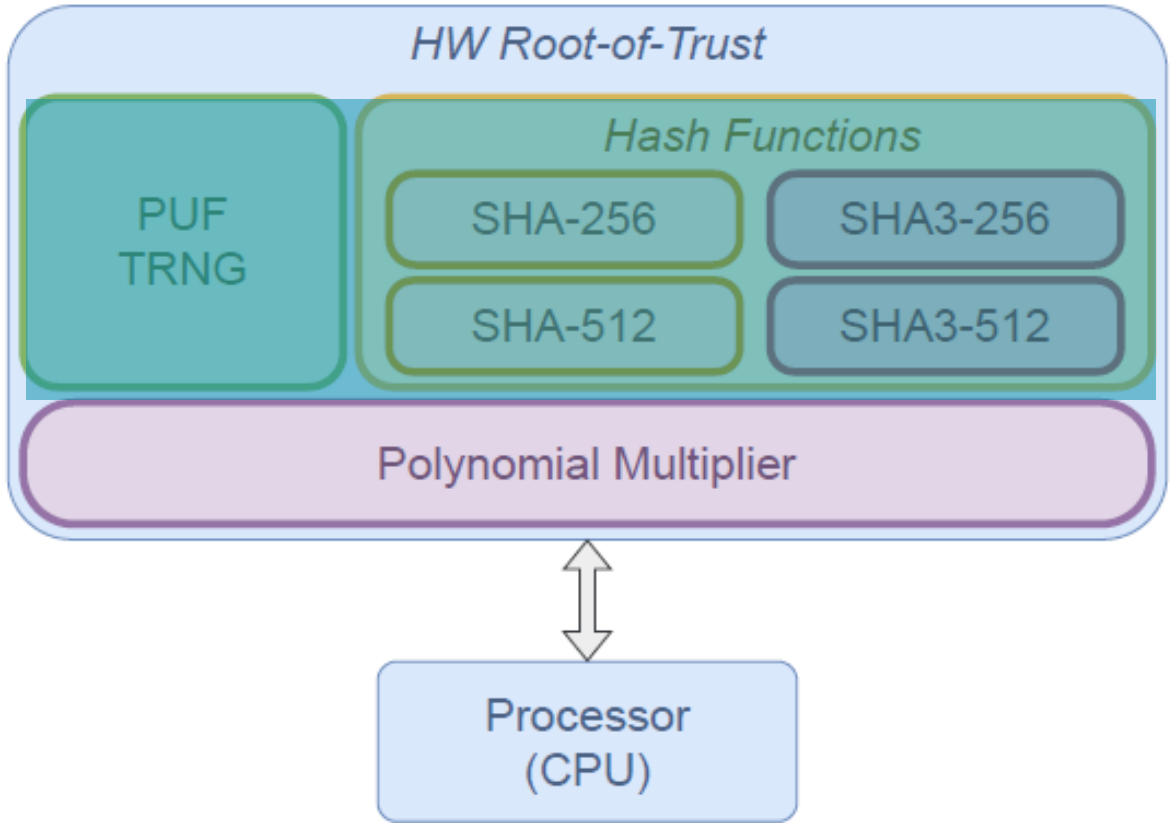
↓

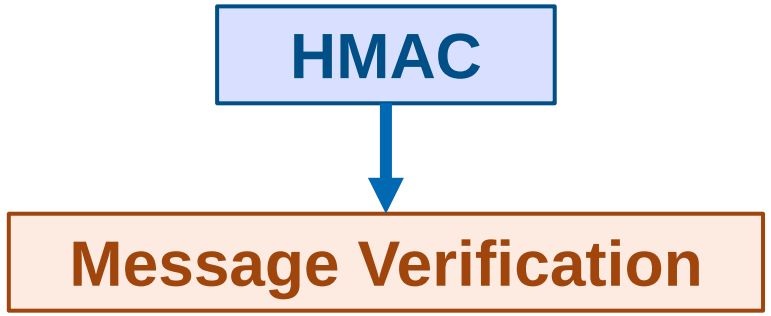
IoT Environment





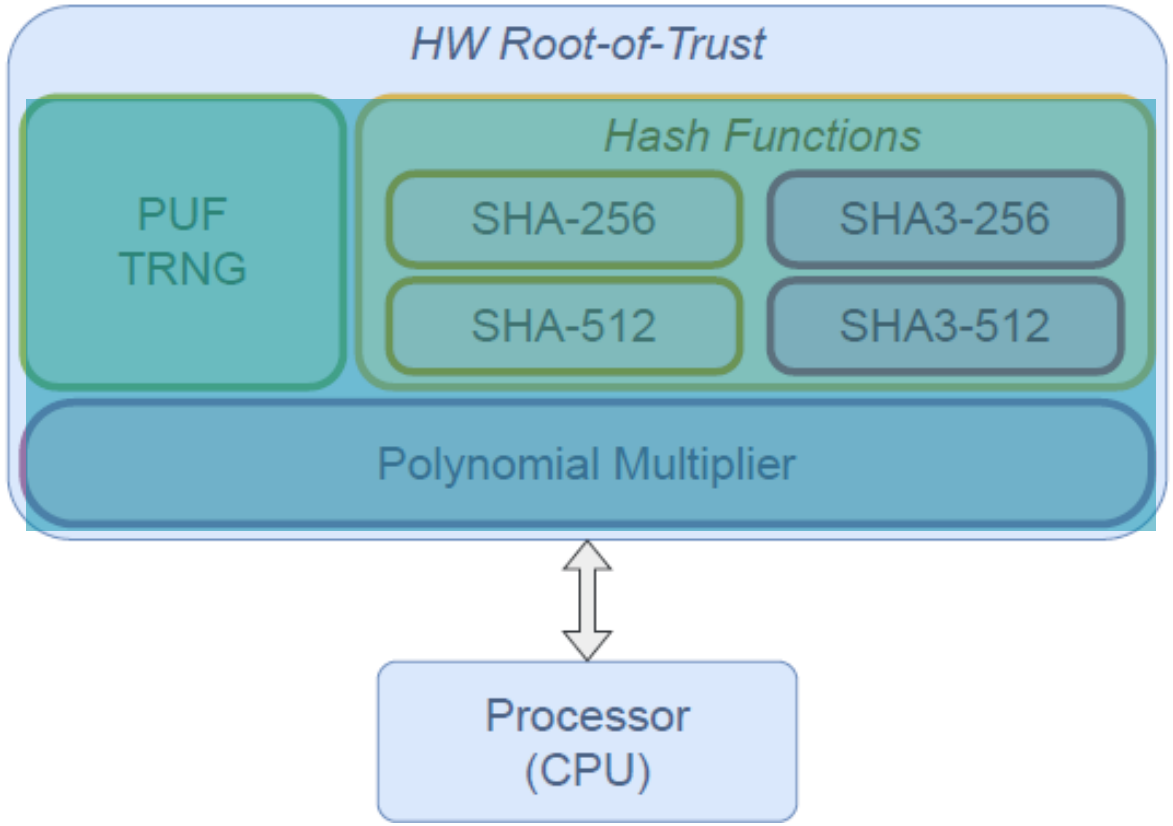
PoC: HMAC






```
root@pynq:/home/xilinx/HMAC_v2# ./HMAC -m 0123456789 -t -s 4

 --- HMAC (SHA3-512) ---
   ---  ID: 64000 ---

 key (l=128):
8b6a3d58f5a47386a10e57042ab7080f

 hmac_hw:
df8cfe26f65eaf693d6631d8bcd2402a9cdf5d8e5e5a1875fc05171f9df06667
0ad40ac4110c2d72605a44dba10a395412bf361008ab750b5f8c48b05f01f984

 hmac_sw:
df8cfe26f65eaf693d6631d8bcd2402a9cdf5d8e5e5a1875fc05171f9df06667
0ad40ac4110c2d72605a44dba10a395412bf361008ab750b5f8c48b05f01f984

 HW Time:        (   154 us.)
 SW Time:        (  1088 us.)
    Acc.:        7.06
```

PoC: NTRU




```
root@pynq:/home/xilinx/NTRU_3Round_DEMO# ./Demo_509 -k -v 1

 --- HARDWARE INITIALIZATION ---

 Generating keys ...
 Generating random seed ...    ---  ID: 64000 --- Complete
 Seed:
0ac419a81eb21924a30f7729a07a2c879e8055dbb30e3ad21c4913c0b4e8ba3a1ec31e7205642d6f66c9fde398a7245c
        Complete
 Storing keys ...        Complete

root@pynq:/home/xilinx/NTRU_3Round_DEMO# ./Demo_509 -e -v 1

 --- HARDWARE INITIALIZATION ---

 Loading public key ...          Complete
 Generating shared secret and ciphertext in SW and HW/SW...
 Performing SHA3-256 in HW ...  Complete
        Complete

     Time SW:    14906 us.
     Time HW:     2208 us.
 Acceleration:   6.750906

 Storing ciphertext ...          Complete
 ss: fba94ea952da1c062cfb1a3d3340f075d49acdd0e6ceb3f055a435b6243eef4e
root@pynq:/home/xilinx/NTRU_3Round_DEMO# ./Demo_509 -d -v 1

 --- HARDWARE INITIALIZATION ---

 Loading secret key ...          Complete
 Loading ciphertext ...          Complete
 Recovering shared secret in SW and HW/SW...
 Performing SHA3-256 in HW ...  Complete         Complete

     Time SW:    45425 us.
     Time HW:    32515 us.
 Acceleration:   1.397048

 ss_SW: fba94ea952da1c062cfb1a3d3340f075d49acdd0e6ceb3f055a435b6243eef4e
 ss_HW: fba94ea952da1c062cfb1a3d3340f075d49acdd0e6ceb3f055a435b6243eef4e
```

NTRU

Improvement of the already presented

Conclusions

The establishment of a hardware RoT significantly bolsters the security of an embedded system. This enhancement is not only robust but also efficient, making it a critical component in the design of secure systems. The considerations for implementation within the framework of the IoT have led to a streamlined design of the RoT. This design is mindful of the constraints of IoT devices, particularly the size of transistors or the occupancy of logic resources in terms of FPGA implementations. The result is a compact yet powerful RoT that fits within the limited resources of IoT devices while providing robust security. In addition to the primary conclusion, there are several other insights extracted from this dissertation. These insights shed light on various aspects of the cryptosystem and contribute to our understanding of its design and implementation. They serve as valuable outcomes that can guide future research and development in this field. These conclusions are:

Conclusions

1. Taking inspiration of the CIA triad as model designed to guide policies for information security, the design of a RoT that encompasses a set of crypto primitives is a suitable solution. The approach taken in this dissertation is primarily from a hardware-oriented perspective. This proposed RoT has included a PUF, which can be used as an identifier generator. They also incorporate hash functions, represented by SHA-2 and SHA-3, which are essential for ensuring data integrity. Moreover, it includes a cryptographic primitive that can securely accelerate the computation of PQC algorithms. One such PQC algorithm is NTRU, which is known for its security and efficiency. The integration of these components within a RoT provides a robust and secure hardware solution for addressing the principles of the CIA Triad in the IoT framework.

Conclusions

2. The initial module developed in this dissertation was a PUF based on a typically undesirable phenomenon known as RTN. Several architectures were proposed in which, basically, the response of the PUF is obtained comparing one metric that encapsulate all the RTN information, the MPF. The initial stages involved designing and evaluating the PUF from a software perspective, which included conducting several simulations. Once positive results were obtained from these simulations, the next phase involved integrating this PUF into an ASIC. This process followed an analog design flow, culminating in the creation of the first silicon-based RTN-PUF, referred to as MILESTONE-I. The successful results of this endeavor led to the filing of an international patent.

Conclusions

3. The following modules developed were those associated with hash functions. After studying the mathematical background of both selected hash functions (SHA-2 and SHA-3), the first step involved implementing the RTL in an FPGA. To this end, a comprehensive set of parameters was incorporated into the integration of these hash functions into IP modules, allowing the implementation of any of the hash versions collected in the standards. One of these designs, specifically SHA-256, was chosen for integration into an ASIC, following a fully digital design flow.

Conclusions

4. The final module that was developed was related to the acceleration of one of the algorithms selected by NIST in the third round of the PQC contest: the NTRU. In that algorithm, the critical component is the polynomial multiplication. To address this, a RTL design was developed, incorporating all the parameter sets outlined in the documentation. Consideration was given not only to performance but also to potential information leakage associated with timing and power attacks. The results demonstrated that it is feasible to enhance the speed of the algorithm while adding an additional layer of security against SCAs.

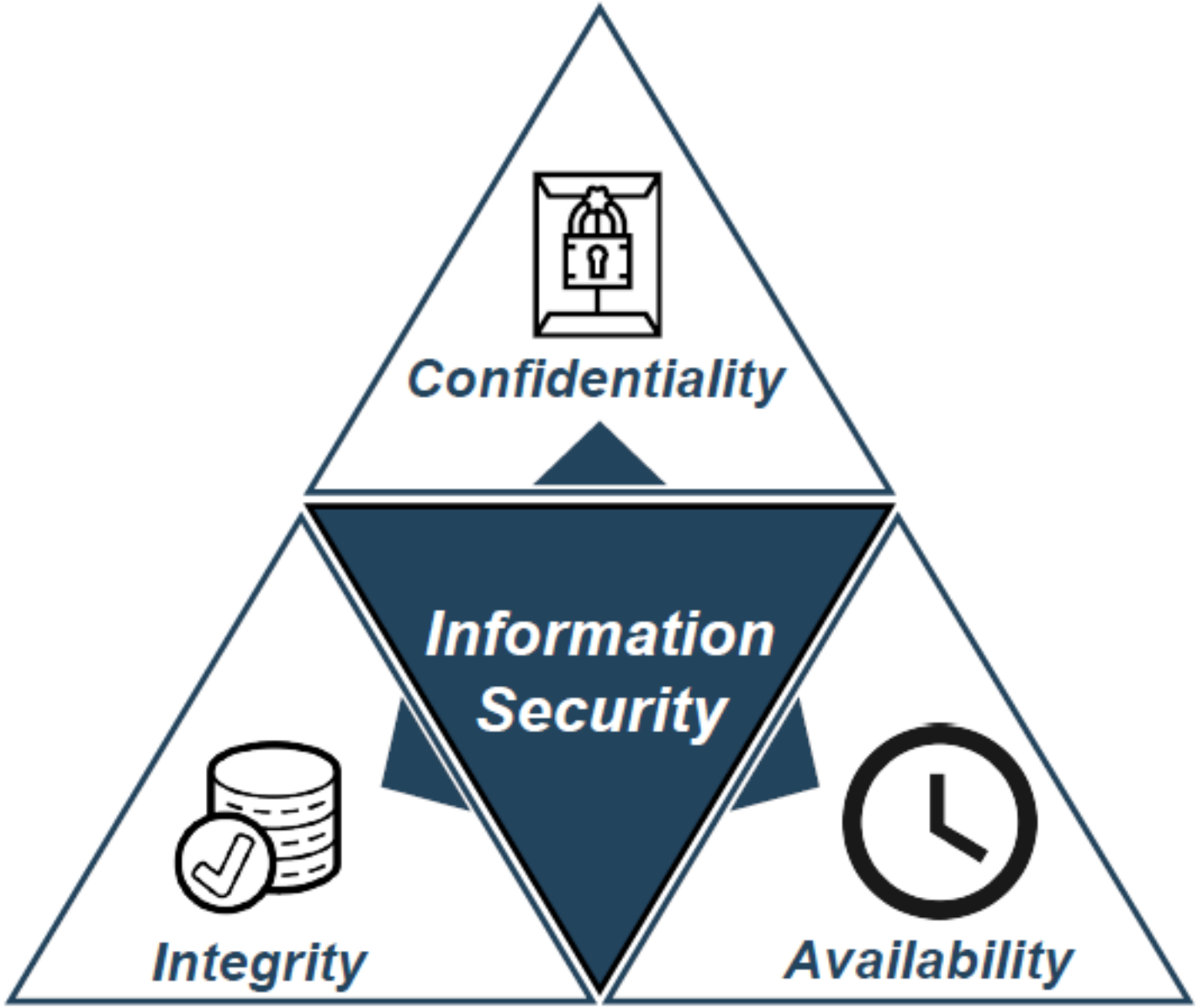
Conclusions

5. Lastly, a final design of the RoT was proposed, integrating all modules into a single hardware platform. This comprehensive design was evaluated in various scenarios, demonstrating the reusability of each module. The results were highly promising, underscoring the effectiveness of this integrated approach in enhancing the security and efficiency of the embedded systems.

IMSE
-cnm

Instituto de
Microelectrónica
de Sevilla

Introduction

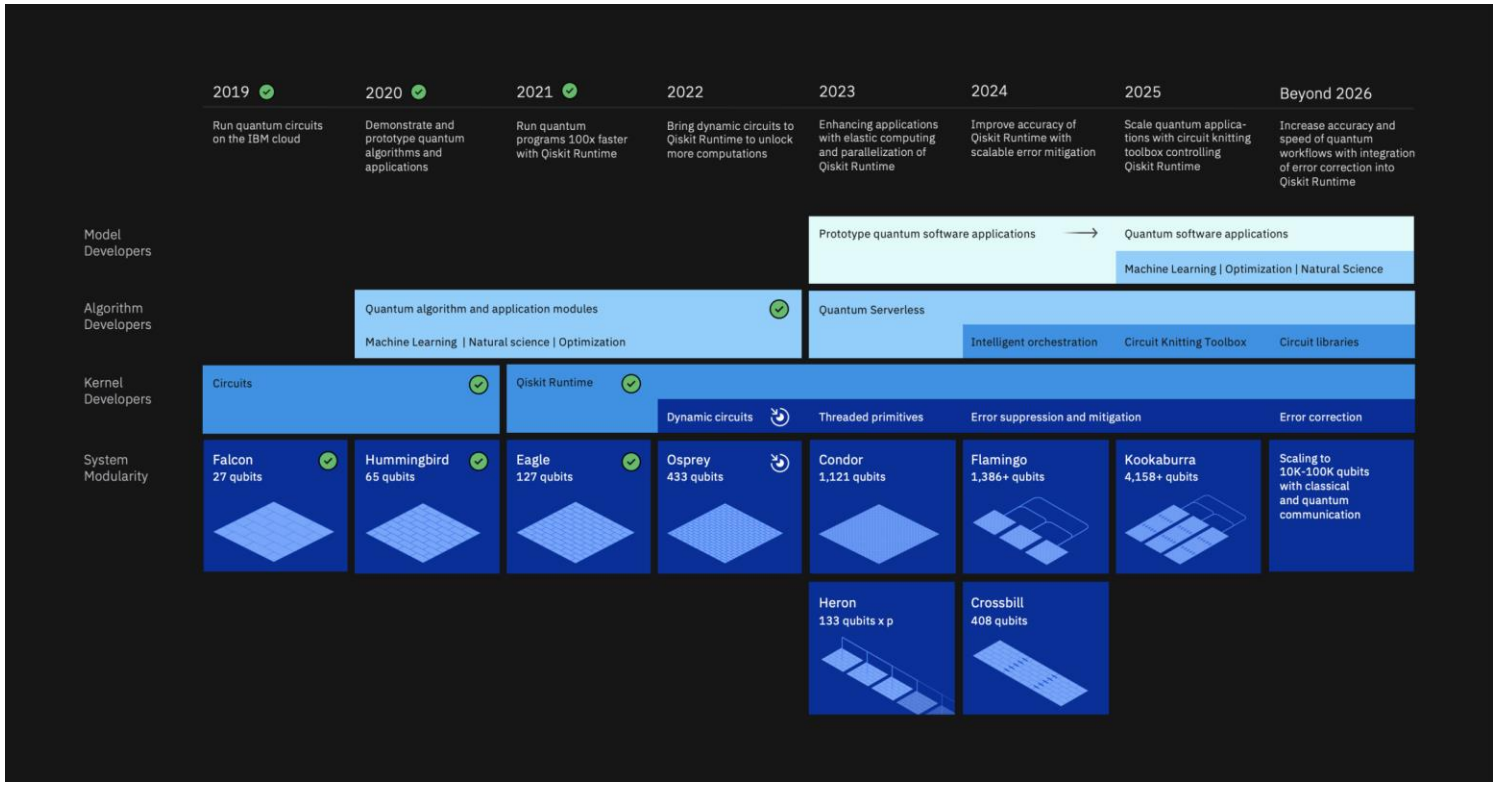


Introduction

https://www.enisa.europa.eu/publications/good-practices-for-security-of-iot-1/@@download/fullReport

https://www.enisa.europa.eu/publications/guidelines-for-securing-the-internet-of-things/@@download/fullReport

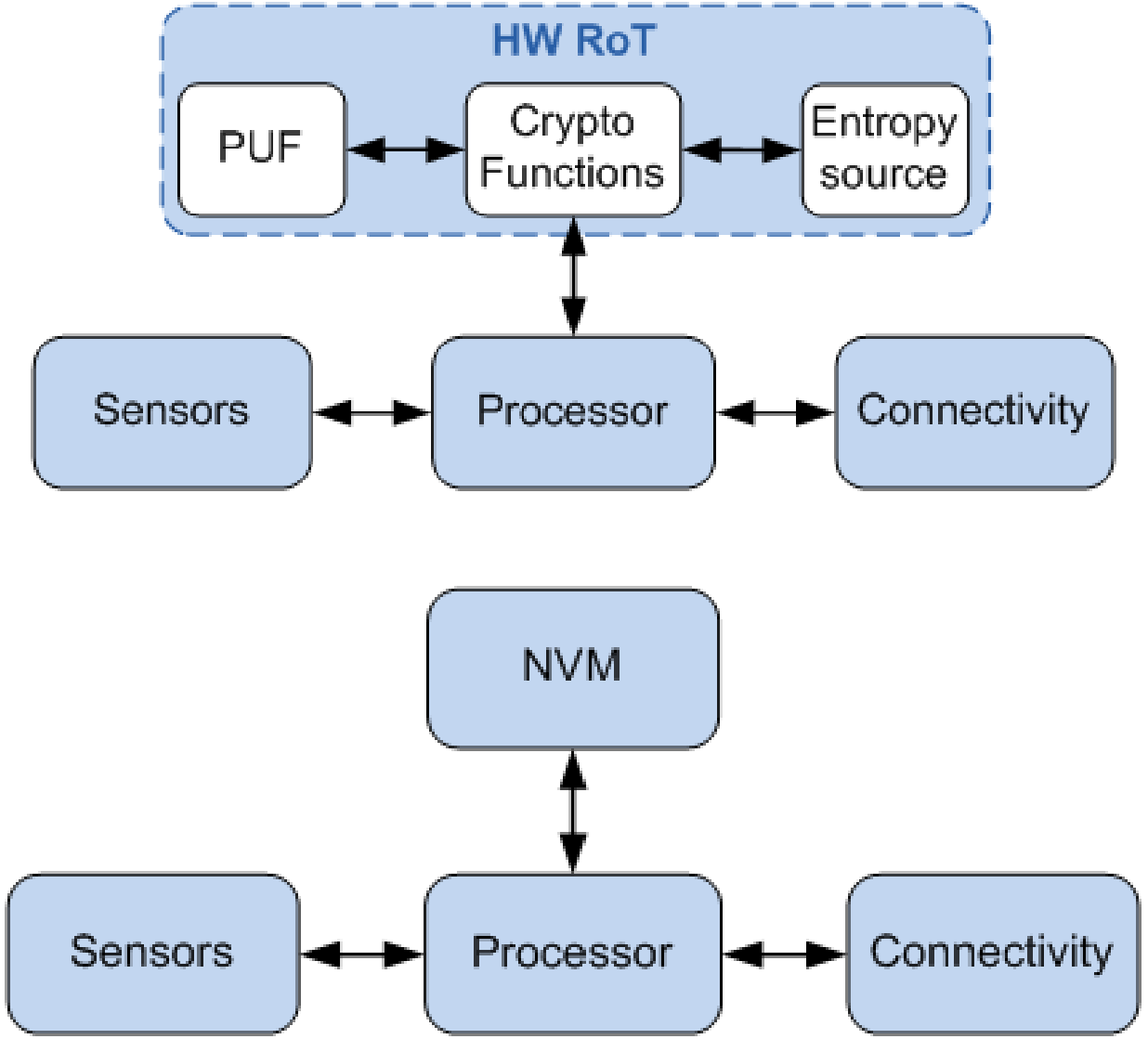
Introduction



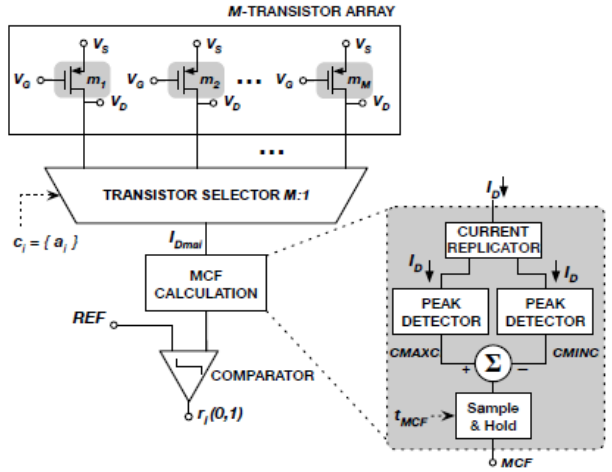
QUBIP

https://qubip.eu/

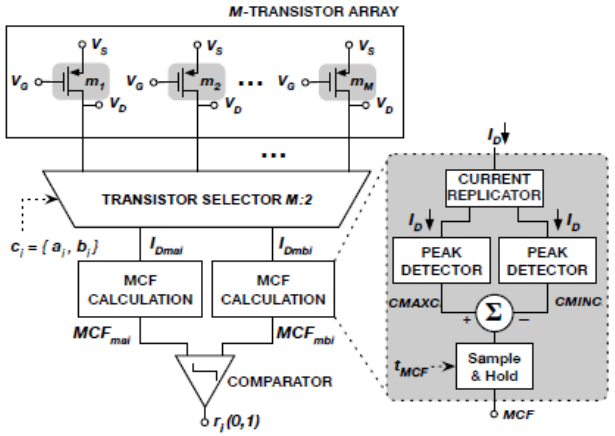
Introduction



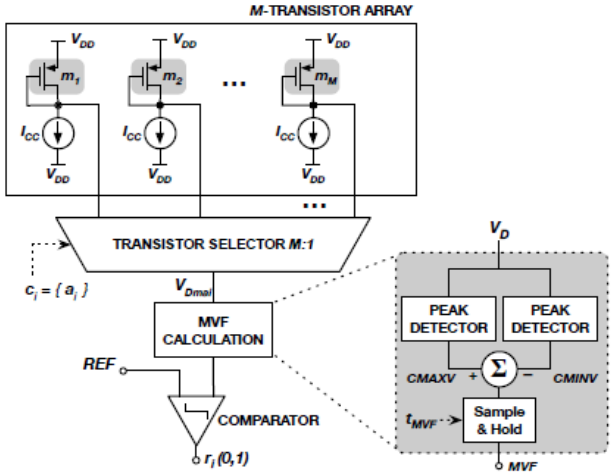
PUF: Architecture



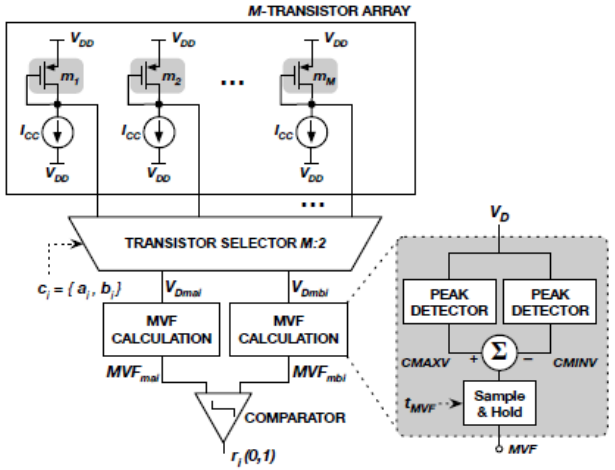
(a) Current-mode configuration comparing with a reference.



(b) Current-mode configuration comparing with another transistor.

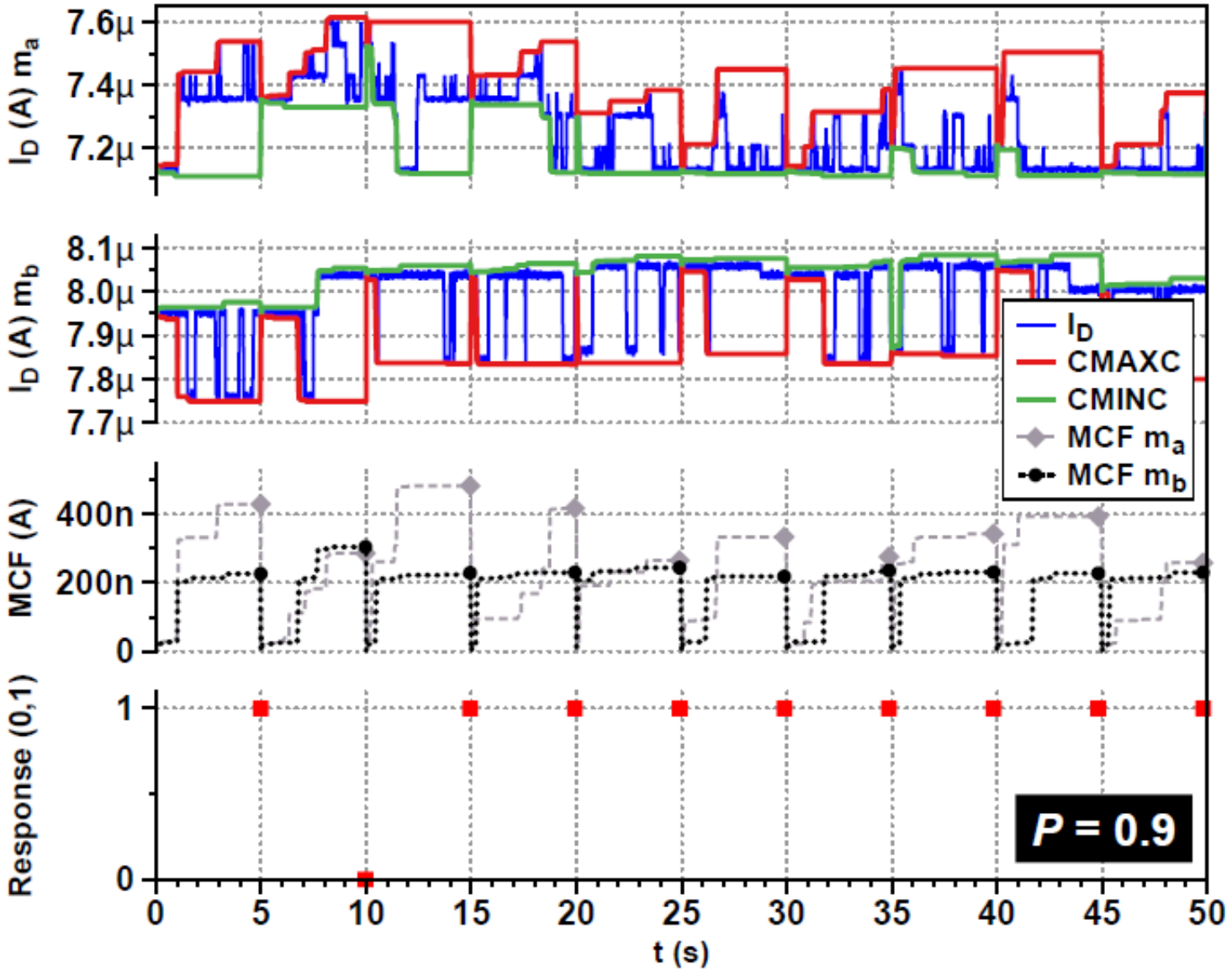


(c) Voltage-mode configuration comparing with a reference.



(d) Voltage-mode configuration comparing with another transistor.

PUF: Bit Selection Method and Metrics



$$HDintra = \frac{1}{m}\sum_{t=1}^{m}\frac{HD(R_i, R'_{i,t})}{n} \times 100\%$$

$$HD(x,y) = \sum_{j=1}^{l} |x_j - y_j|$$

$$HDinter = \frac{2}{k(k-1)}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\frac{HD(R_i, R_j)}{n} \times 100\%$$

$$HW = \frac{1}{n}\sum_{l=1}^{n} r_{i,l} \times 100\%$$