Conference for Next Generation Arithmetic
Illinois, Chicago, USA
14 November 2026

CoNGA

Securing Posit Arithmetic Units using Logic Obfuscation for Hardware Integrity Protection

by

Diksha Shekhawat[1,2], Jugal Gandhi[1,2], M. Santosh[1,2], Jai Gopal Pandey[1,2], Jaya Dofe[3], John L. Gustafson[4], and Chandra Shekhar[5]

[1] CSIR- Central Electronics Engineering Research Institute (CEERI), Pilani- 333031, India

[2] Academy of Scientific and Innovative Research (AcSIR), Ghaziabad- 201002, India

[3] California State University, Fullerton, CA 92831-3599, USA

[4] Arizona State University (ASU), Arizona- 85287, USA

[5] Birla Institute of Technology and Science (BITS) Pilani, Pilani-333031, India

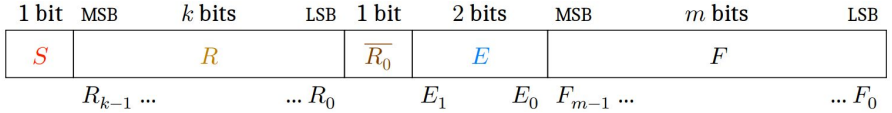


Outline

- Emergence of Posit Number System
- Hardware Accelerators for Posit Computing
- IC Design Chain and Need for IP Security
- Motivation for the Work
- Introduction to Logic Locking
- The Proposed Work
- Experimental Evaluation and Results
- Conclusion
- References

Posit Number System

- Format: sign, regime, exponent, and fraction [GUST17]



Generic posit representation with all fields

- Regime length (r) based on R_0 (regime check bit) consists of k
  - R_0 = 0, then r = -k; R_0 = 1, then r = k-1 [GUST22]

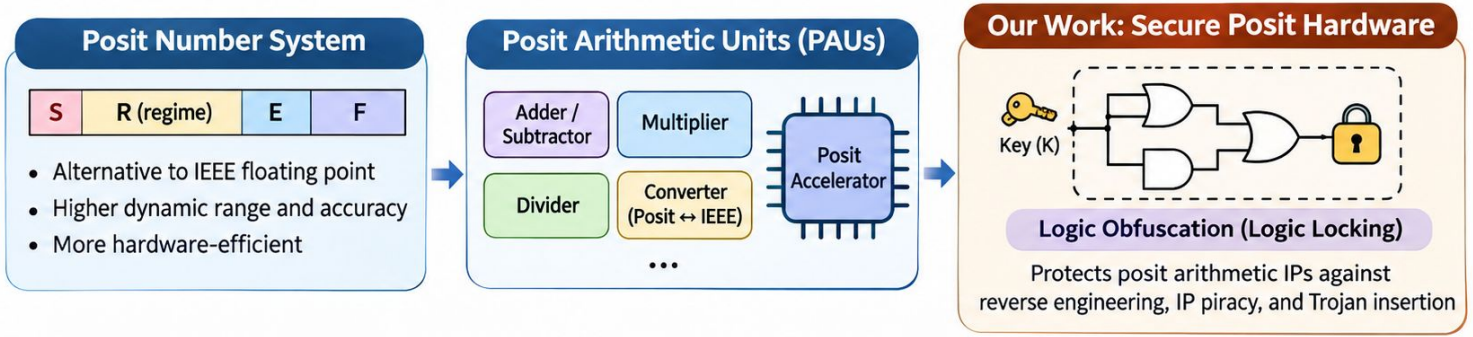
Binary	0000	0001	001x	01xx	10xx	110x	1110	1111
Numerical meaning, *k*	-4	-3	-2	-1	0	1	2	3

- Posit representation

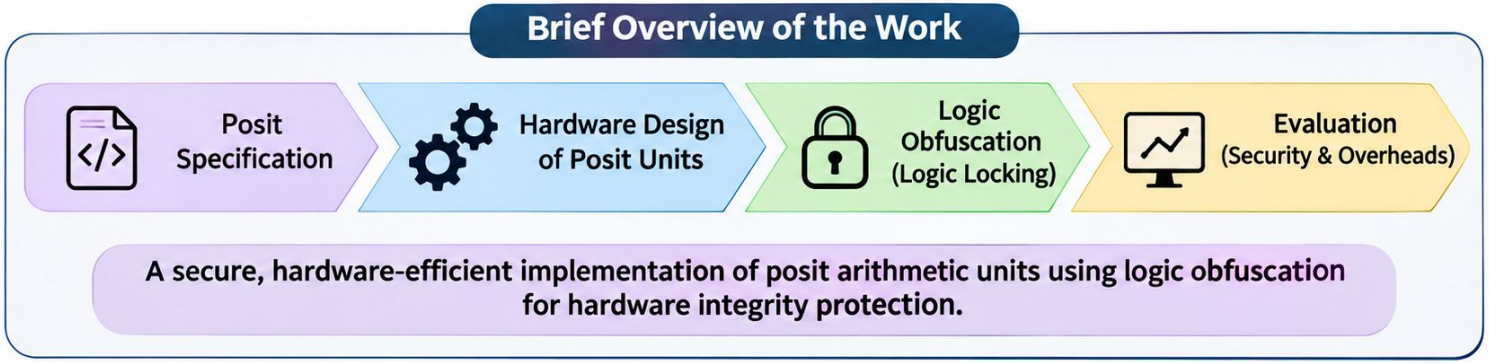
$$P = (1-3s)+f) \times 2^{(1-2s)\times(4r+e+s)} \; ; es = 2 \quad \ldots \quad (1)$$

[GUST17]: J. L. Gustafson and I. T. Yonemoto, "Beating Floating Point at its Own Game: Posit Arithmetic," Supercomputing frontiers and innovations, vol. 4, no. 2, pp. 71-86, 2017.
[GUST22]: Posit Working Group, "Standard for Posit Arithmetic (2022)," 2022. https://posithub.org/docs/posit_standard-2.pdf.

Hardware Development using Posit



Brief Overview of the Work

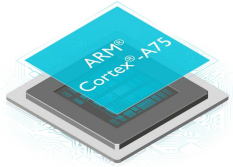


Need for Hardware Security in Integrated Circuits

- Hardware-level vulnerabilities are hard to fix post-fabrication
  - Software patches cannot address silicon-level flaws
- Hardware-level attacks
  - *Specter* and *Meltdown*
  - Reset Glitch
- Still Intel's 230+ CPU models 10 families are silicon vulnerable









MICROSOFT SECURITY XBOX HACKING

Hacker unveils exploit that cracks the "unbreakable" Xbox One at the silicon level

The technique hijacks the console's boot process and bypasses Microsoft's security architecture

By Skye Jacobs March 16, 2026 at 7:04 AM

Meltdown and Spectre CPU Flaws Affect Intel, ARM, AMD Processors

The Hacker News Jan 04, 2018

SECURITY

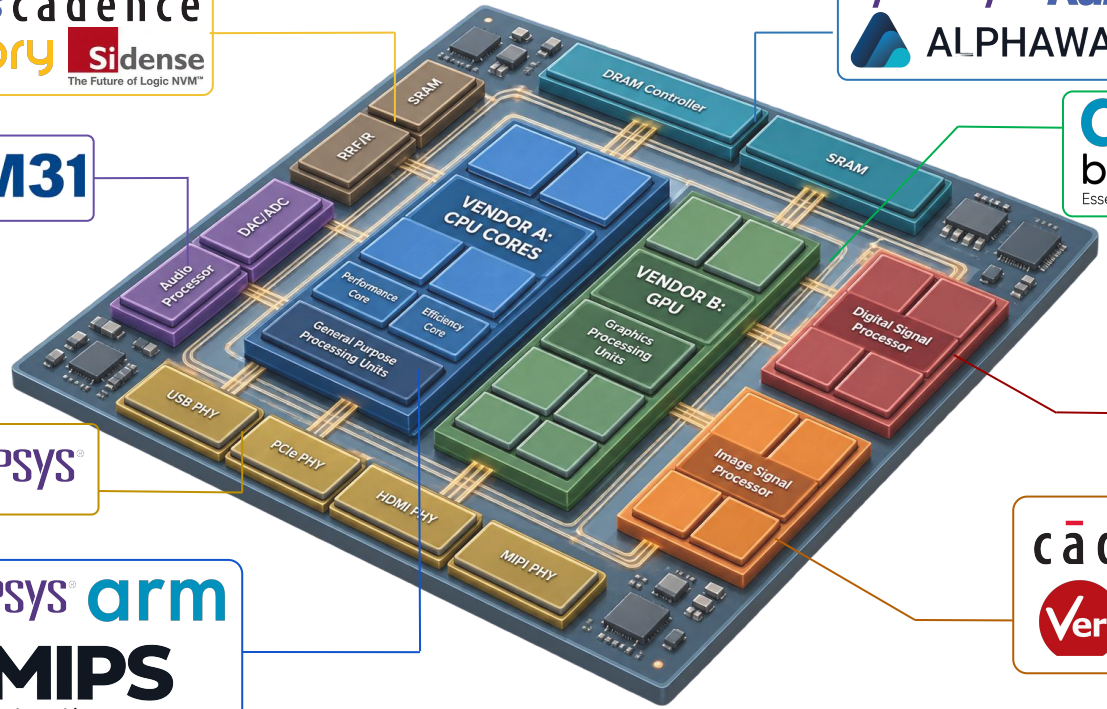
Intel admits a load of its CPUs have Spectre v2 flaw that can't be fixed

And won't fix Meltdown *nor* Spectre for 10 product families covering 230-plus CPUs

Simon Sharwood

Source: https://thehackernews.com/2018/01/meltdown-spectre-vulnerability.html
https://www.theregister.com/security/2018/04/04/intel-admits-a-load-of-its-cpus-have-spectre-v2-flaw-that-cant-be-fixed/322262

Distributed System-on-Chip Design



Rambus cadence ememory Sidense The Future of Logic NVM™

REALTEK M31

mixel Mixed-Signal Excellence T2M IP SYNOPSYS

cadence SYNOPSYS arm SiFive MIPS a GlobalFoundries company

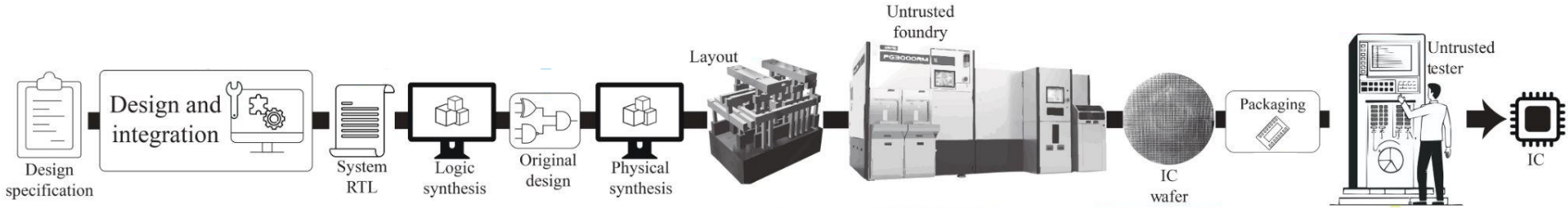
SYNOPSYS Rambus openfive ALPHAWAVE SEMI SAMSUNG

arm brainchip Essential AI VeriSilicon

SYNOPSYS CEVA arm QuickLogic LATTICE SEMICONDUCTOR

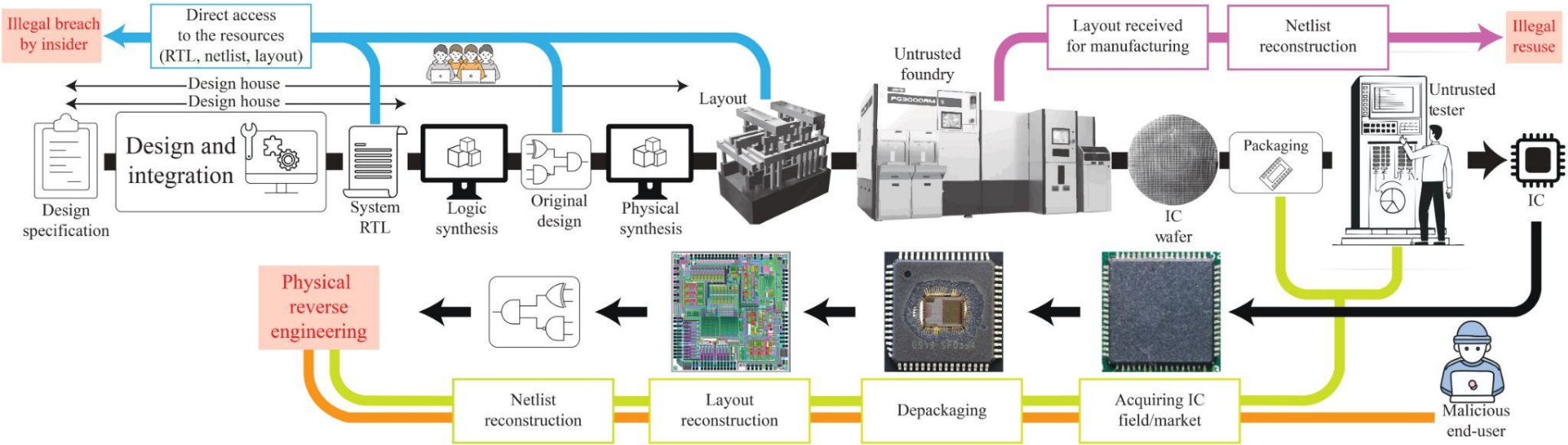
cadence SYNOPSYS VeriSilicon altek

Security Threats in the IC Design Flow



IC: Integrated circuit; RTL: Register transfer level

Security Threats in the IC Design Flow



IC: Integrated circuit; RTL: Register transfer level

Design Threats

Logical
Extract data flow weaknesses



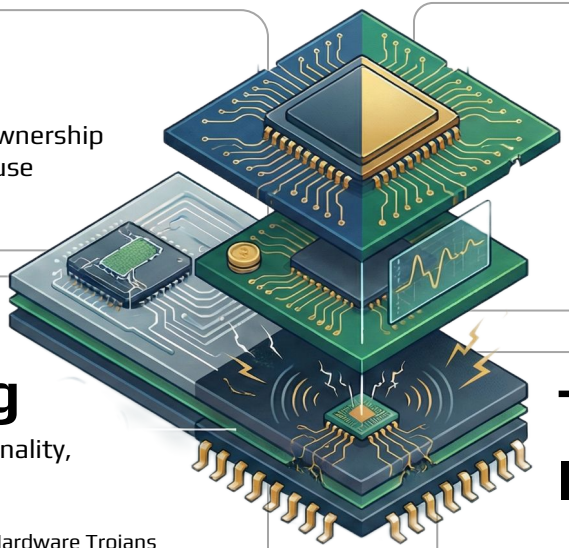


Physical
Monitor physical patterns



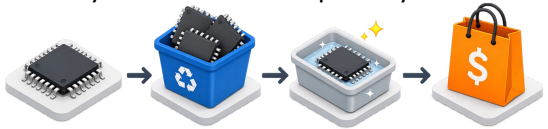
IP Piracy

- Copy/Steal and claim ownership
- Illegal or unlicensed IP use

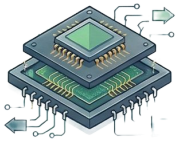


Counterfeiting

- Reliability issues of the complete system



Scrapped ICs → Repackaged, Collected → Polished → Sold as *new* at cheaper rates

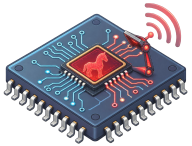


Reverse Engineering

- Extract design, functionality, and technology details
- Work as enabler for
  - IP piracy or insertion of Hardware Trojans

深圳思驰科技有限公司
Shenzhen sichi technology co.,LTD.
WhatsApp: +86 187 0761 9926
+86 189 0285 6696
PCB Reverse,PCB Copy,PCB Clone,PCB Fabrication,PCBAssembly,PCB Manufacturing,PCB Designing,PCB Layout; MCU Reverse,MCU Attack,IC Crack,IC deciphering
Email:sales@pcbic-reverse.com

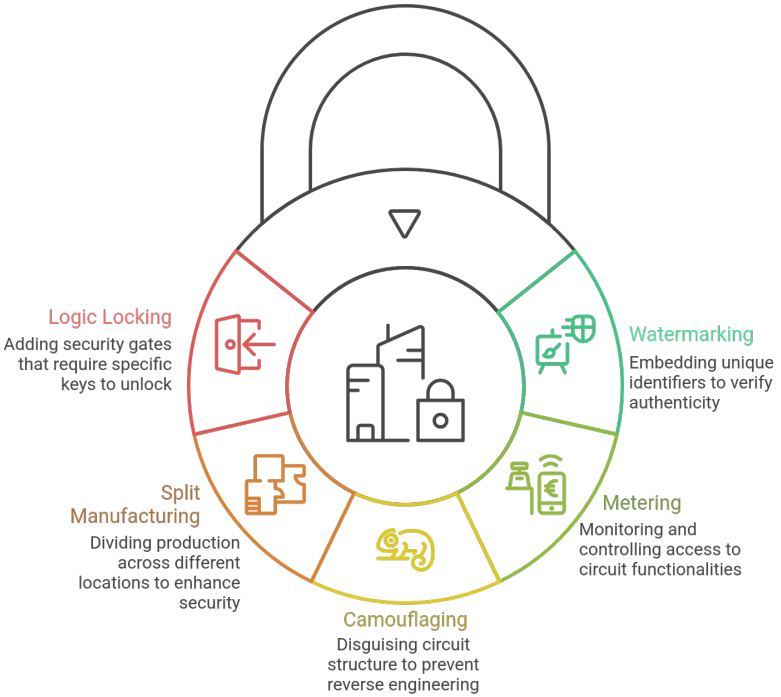
Trojan Insertion



- Trojans try to bypass or disable system security
- Leaking confidential information
- Disable or damage chip or components of it

IP: Intellectual property

Design-for-Trust (DfTr) Techniques



3PIP: Third party IP

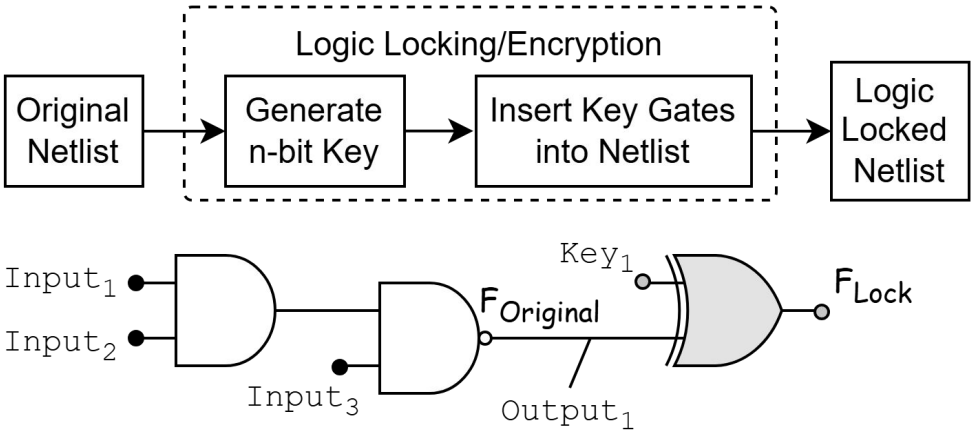
Design-for-Trust	Untrusted Entities			
	3PIP Vendors	Foundry	Test Facility	End-user
Watermarking	✗	✗	✗	✗
Camouflaging	✗	✗	✗	✓
Split Manufacturing	✗	✓	✗	✗
Metering	✗	✗	✓	✓
Logic Locking	✓	✓	✓	✓

✓ denotes that technique can protect piracy conducted by an untrusted entity
✗ denotes the inability to defend against attacks

Logic locking/obfuscation as most prominent technique for secure hardware design

Introduction to Logic Locking

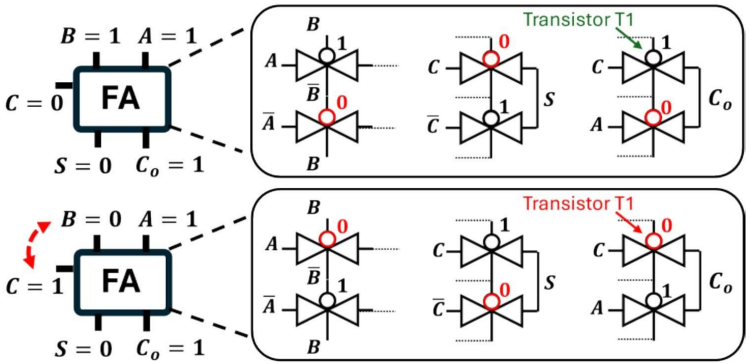
- Hardware security technique that protect logic within ICs
- Security logic controlled by key stored in a nonvolatile memory
- Obfuscating netlist functionality



Key	Input$_1$	Input$_2$	Input$_3$	$F_{Original}$	F_{Lock}
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	0	0
0	0	1	1	0	0
0	1	0	0	0	0
0	1	0	1	0	0
0	1	1	0	0	0
0	1	1	1	1	1
1	0	0	0	0	1
1	0	0	1	0	1
1	0	1	0	0	1
1	0	1	1	0	1
1	1	0	0	0	1
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	1	0

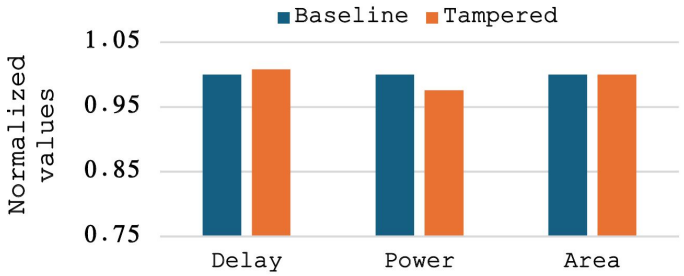
Shortcomings in Existing Hardware Design

Functionally equivalent rewiring



Swapping full-adder inputs (B ↔ C) keeps sum and carry unchanged but moves NBTI stress onto PMOS T1.

Invisible to conventional checks



Delay and power differ by only 0.8% and 2.5%; area is unchanged, so post-manufacturing tests pass.

Why it matters

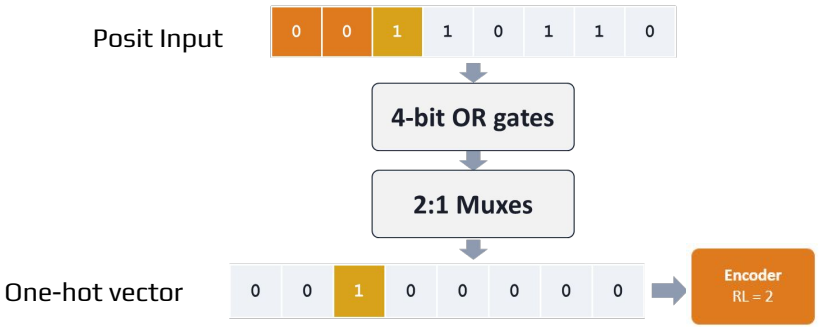
An untrusted foundry or third-party IP vendor can *tamper* the arithmetic netlist before fabrication.

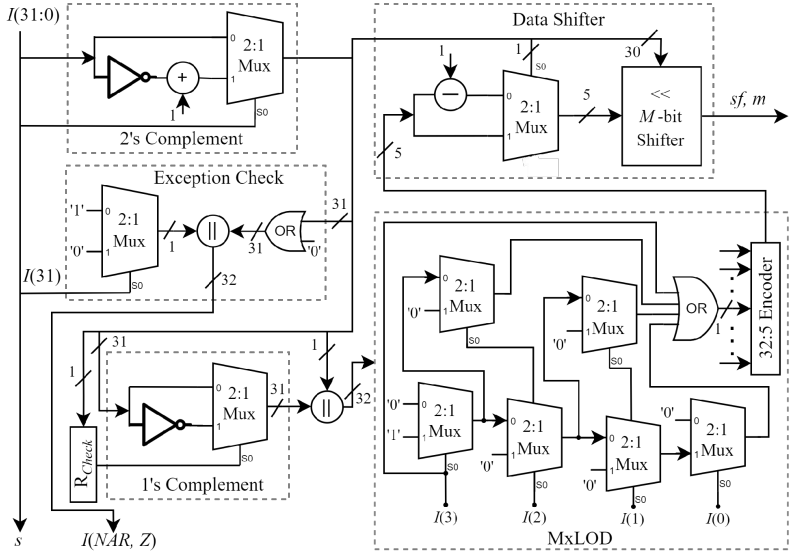
Arithmetic IP needs design-time protection that hides its structure and function from such parties.

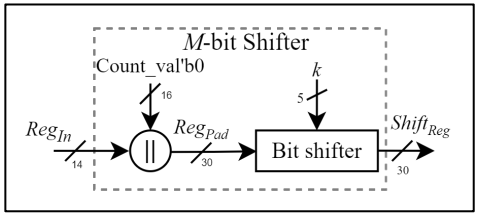
Source: M. Heidary and B. K. Joardar, "Attacking AI Accelerators via Arithmetic Properties," Journal of Hardware and Systems Security, 10:19, 2026

Posit Data Decoder

- Posit data decoder
  - Utilizes a multiplexer-based LOD
- Data Shifter
  - Barrel shifter
  - *M*-bit shifter

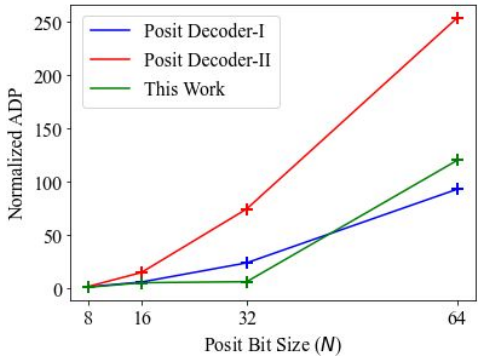


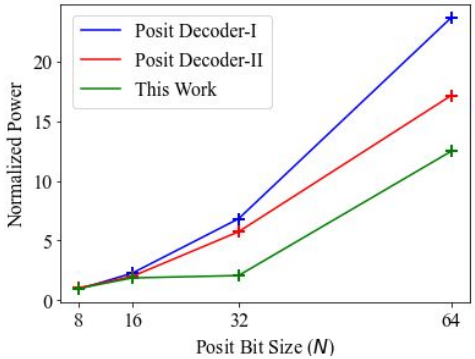


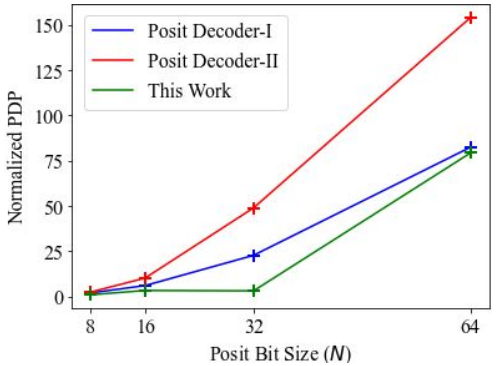


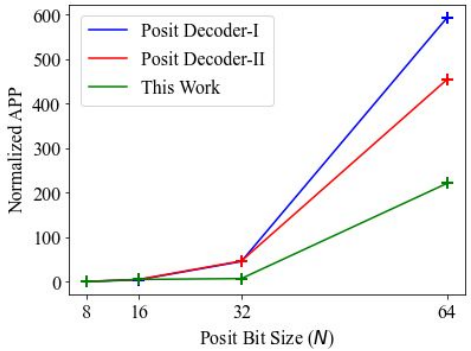
[DSH26] D. Shekhawat, *et al.* "Parameterized Decoder Architecture with One-hot Regime Extraction for Posit Arithmetic Units."IEEE 19th International MCSoC, Shanghai, China, 2026. (Accepted)

Results: ASIC Implementations and Comparison









The proposed approach have overall better ADP, normalized power, PDP, and APP

Technology node: 32 nm CMOS
ADP: Area-delay product; PDP: Power-delay product; APP: Area-power product
Posit Decoder-1: LDD-based decoder; Posit Decoder-2: LOD-based decoder

[DSH26] D. Shekhawat, *et al.* "Parameterized Decoder Architecture with One-hot Regime Extraction for Posit Arithmetic Units."IEEE 19th International MCSoC, Shanghai, China, 2026. (Accepted)
[SUN24] J. Sun and Y. Lao, "Efficient Data Extraction Circuit for Posit Number System: LDD-Based Posit Decoder," IEEE TCAD, 2023.

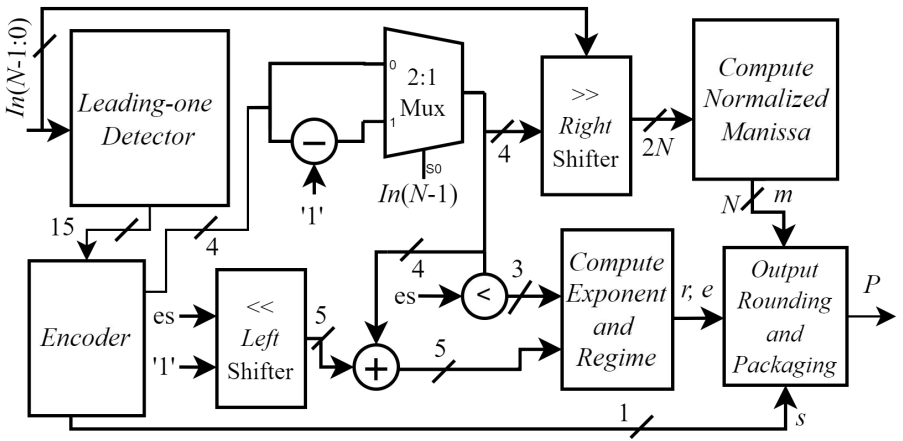
Posit Conversion Architecture

- Integer/Real to Posit conversion:

Algorithm 1: Int/Real-to-Posit Conversion

Input : $\mathcal{I}n$: Input value
Output: $\mathcal{P}$: Posit output

Given: $\mathcal{N}$: `Bit width`, *es*: `Exponent size`
$\mathcal{P}_{val} \leftarrow$ **LOD**($\mathcal{I}n$[$\mathcal{N}$-2:0]) ▷ Leading-one detection
if $\mathcal{I}n$[$\mathcal{N}$-1:$\mathcal{N}$-$\mathcal{P}_{val}$]=1 **then**
  $\mathcal{T} \leftarrow \mathcal{P}_{val}$
else
  $\mathcal{T} \leftarrow \neg\, \mathcal{P}_{val}$
end
$\mathcal{T}_{emp} \leftarrow [(1 \ll es) + \mathcal{T}]$
$\mathcal{R}_{length} \leftarrow$ **Max** (es, $\mathcal{T}$)
$\mathcal{C}_{ount} \leftarrow \mathcal{T}_{emp}$ [$\mathcal{R}_{length}$:es]
$\mathcal{E}xp \leftarrow \mathcal{T}_{emp}$ [es-1:0] ▷ Exponent computation
$\mathcal{R}egime$[0] $\leftarrow$ 0
for *i in range 1 to* $\mathcal{C}ount$ **do**
  $\mathcal{R}egime$ [$\mathcal{C}ount$:i] $\leftarrow$ 1 ▷ Regime computation
end
$\mathcal{IN}_{shifted} \leftarrow \mathcal{I}n \gg \mathcal{T}$
$\mathcal{M} \leftarrow \mathcal{IN}_{shifted}$ [$\mathcal{N}$-1:es + $\mathcal{C}ount$ + 2]
Output $\mathcal{P} \leftarrow \{s, \mathcal{R}egime, \mathcal{E}xp, \mathcal{M}\}$



[DSH25a] D. Shekhawat, *et al.* "Parameterized Low-overhead Data Conversion Architectures for Posit Arithmetic."IEEE 68th International MWSCAS, Michigan, USA, 2025.

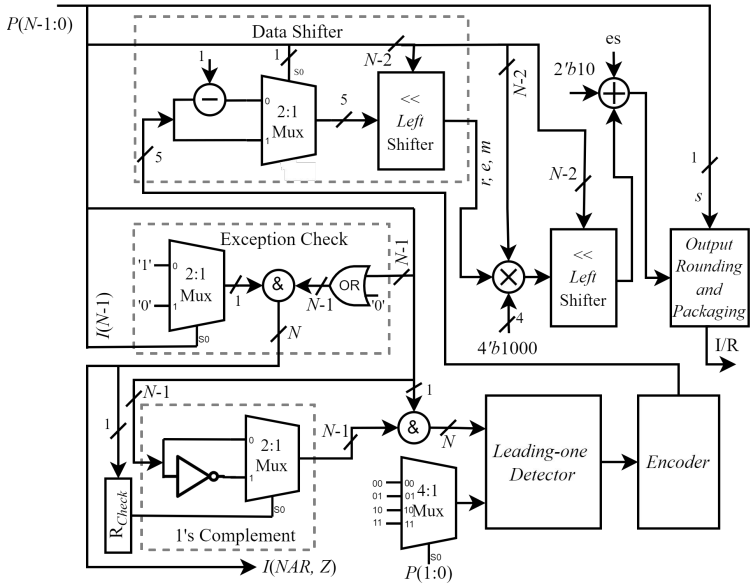
Posit Conversion Architecture...

- Posit to integer/real conversion: $P = ((1-3s) + f) \times 2^{(1-2s)\times(4r+e+s)}$

Algorithm 2: Posit-to-Int/Real Conversion

Input : $\mathcal{P}$: Input posit value
Output: $\mathcal{I}/\mathcal{R}$: Int/Real output

Given: $\mathcal{N}$: `Bit width`, *es*: `Exponent size`
Data Extraction($\mathcal{P}$, s, $\mathcal{R}_{check}$, $\mathcal{K}$, e, m)
$\mathcal{R}_{check} \leftarrow \mathcal{P}[\mathcal{N}$ - 1] ▷ Regime check bit
$\mathcal{K}$ computes using Equation (2)
$\mathcal{XIN} \leftarrow \mathcal{P}$ [$\mathcal{N}$-2 : 0]
$\mathcal{K}_{Count} \leftarrow [\mathcal{N}$ - 1] - $\mathcal{K}$
$e_1 \leftarrow \mathcal{P}$ [$\mathcal{N}$-$\mathcal{K}_{Count}$: *es*] ▷ Exponent initialization
if $\mathcal{P}[\mathcal{N}$-2 : 0]=1 **then**
  $e_2 \leftarrow$ 4'b1000 × [$\mathcal{K}_{Count}$ - 1]
else
  $e_2 \leftarrow \neg$ (4'b1000 × [$\mathcal{K}_{Count}$)] + 1'b1
end
if e_2=1 **then**
  $e_f \leftarrow e_1 + e_2$
else
  $e_f \leftarrow \neg (e_1 + e_2)$ + 1'b1
end
Output $\mathcal{I}/\mathcal{R} \leftarrow s \times [(\mathcal{XIN} \ll \mathcal{K}_{Count}) + e_f$ + 2'b10]



[DSH25a] D. Shekhawat, *et al.* "Parameterized Low-overhead Data Conversion Architectures for Posit Arithmetic."IEEE 68th International MWSCAS, Michigan, USA, 2025.

Results: ASIC Implementations and Comparison

Work	Design	Posit Bit Size (*N*)	28 nm			45 nm		
			Area (μm²)	Power (μW)	Delay (ns)	Area (μm²)	Power (μW)	Delay (ns)
[DMA22]	I2P	32	175.32	63.07	1.47	493.96	216.93	14.36
This Work			60.36	15.01	0.46	163.59	46.29	3.91
[DMA24]		64	154.45	55.74	1.05	424.53	193.83	10.07
This Work			412.18	101.28	1.23	557.26	165.47	13.63
[DMA22]	P2I	32	194.82	123.28	1.38	528.28	371.70	12.33
This Work			162.19	58.92	0.76	432.78	194.29	8.10
[DMA24]		64	223.24	112.74	1.56	595.57	371.70	12.33
This Work			198.84	42.38	1.01	557.26	120.57	10.68
	R2P	16	125.73	37.97	0.67	346.33	133.12	6.99
		32	272.53	76.13	0.94	715.80	259.27	9.99
	P2R	16	276.45	141.65	1.16	751.45	456.69	12.74
		32	475.39	241.35	1.60	1248.00	728.48	16.85
	I2P	16	15.77	4.56	0.24	46.55	14.83	2.45
	P2I		23.03	5.72	0.25	59.58	16.67	2.81

- The I2P design has shown improvement
  - 65.57% in area
  - 68.71% in delay
  - 76.20% in power
- The P2I design has shown improvement
  - 16.75% in area
  - 44.93% in delay
  - 52.91% in power

For exponent size (*es*)=2

I2P: Integer to Posit; P2I: Posit to integer; R2P: Real to Posit; P2R: Posit to real;

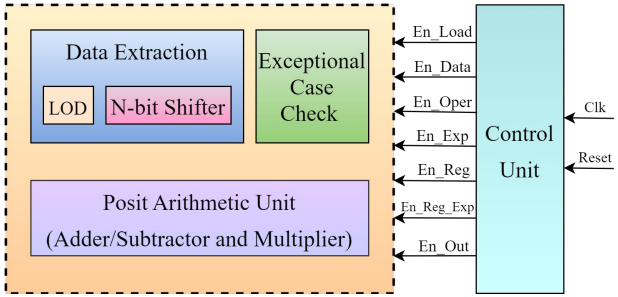
[DSH25a] D. Shekhawat, *et al.* "Parameterized Low-overhead Data Conversion Architectures for Posit Arithmetic."IEEE 68th International MWSCAS, Michigan, USA, 2025.

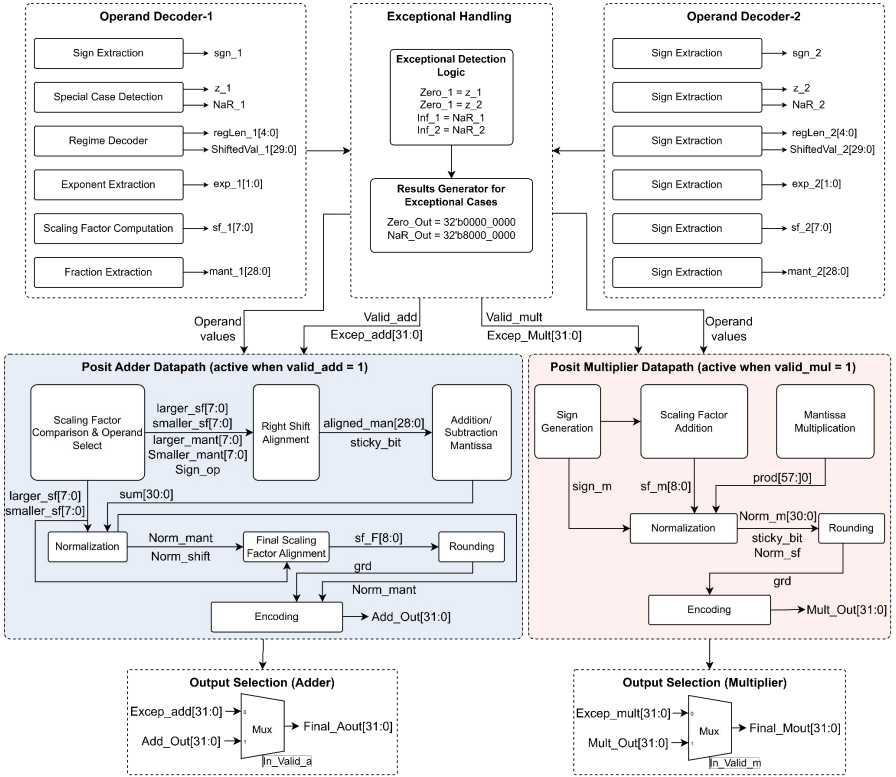
[DMA22] D. Mallasen, *et al.* "PERCIVAL: Open-source Posit RISC-V Core with Quire Capability," IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 3, pp. 1241–1252, 2022

[DMA24] D. Mallasen, *et al.* "Big-PERCIVAL: Exploring the Native Use of 64-Bit Posit Arithmetic in Scientific Computing," IEEE Transactions on Computers, 2024.

Posit Adder and Multiplier Units

- Addition/subtraction and multiplication
- 32-bit architectures with different es with control unit
- Control unit enables the required components





[DSH23] D. Shekhawat, *et al.* "PHAc: Posit Hardware Accelerator for Efficient Arithmetic Logic Operations." Conference on Next Generation Arithmetic, Tampines, Singapore, 2023.

Results: Performance Comparison

Comparison of datapath delay for 32-bit posit adder and multiplier

Bit Pattern (*N,es*)	Posit Adder			Posit Multiplier		
	[JAIS19]	[DIKSH21]	This Work	[JAIS19]	[DIKSH21]	This Work
(32,1)	19.150	27.703	**7.534**	19.076	28.025	**9.310**
(32,2)	19.984	29.009	**6.896**	18.921	27.982	**9.310**
(32,3)	19.752	27.454	**6.389**	19.203	29.850	**8.453**

Comparison of area-delay (AD) product for 32-bit posit adder and multiplier

Bit Pattern (*N,es*)	Posit Adder			Posit Multiplier		
	[JAIS19]	[DIKSH21]	This Work	[JAIS19]	[DIKSH21]	This Work
(32,1)	23190.65	29337.45	**11956.45**	12742.77	19953.80	**4180.19**
(32,2)	25480.00	31068.63	**11109.46**	12866.28	19503.45	**4170.88**
(32,3)	22155.00	29650.32	**10439.62**	13826.20	21103.95	**3753.12**

- Datapath delay
  - Adder □ 64.64%
  - Multiplier □ 52.66%
- Area-delay product
  - Adder □ 52.69%
  - Multiplier □ 69.30%

[DSH23] D. Shekhawat, *et al.* "PHAc: Posit Hardware Accelerator for Efficient Arithmetic Logic Operations." In CoNGA, pp. 88-100. Cham: Springer Nature Switzerland, 2023.
[JAIS19] Jaiswal, M. K., & So, H. K. H. (2019). PACoGen: A hardware posit arithmetic core generator. IEEE access, 7, 74586-74601.
[DIKSH21] Shekhawat, D., et. al. A hardware generator for posit arithmetic and its FPGA prototyping. In 2021 25th International Symposium on VDAT (pp. 1-6). IEEE.

Posit-based MAC

- (16×*N*)-bit quire is implemented as two (8×*N*)-bit Brent–Kung stages with a carry-save (CSA) front end
  - Reduces critical path delay, routing complexity by parallelizing addition

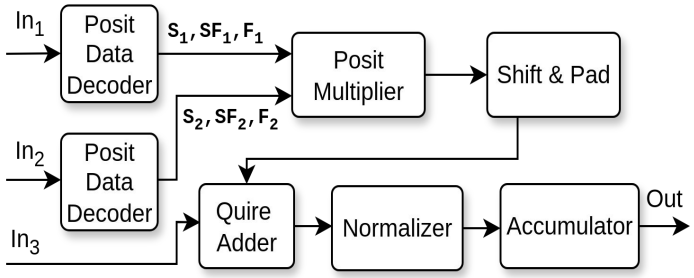
$$(\mathtt{Q}'_s, \mathtt{Q}'_c) = \mathcal{CSA}(\mathtt{Q}_s, \mathtt{Q}_c, \mathtt{Q}_p)$$

$$\mathtt{BK}_U = \mathcal{BK}(\mathtt{Q}'_s[16 \times \mathcal{N} - 1 : 8 \times \mathcal{N}], \mathtt{Q}'_c[16 \times \mathcal{N} - 1 : 8 \times \mathcal{N}])$$

$$\mathtt{BK}_L = \mathcal{BK}(\mathtt{Q}'_s[8 \times \mathcal{N} - 1 : 0], \mathtt{Q}'_c[8 \times \mathcal{N} - 1 : 0])$$

- CSA minimizes carry propagation
  - Enables faster and hardware-efficient accumulation

$$\mathtt{Out} = \mathcal{R}(\mathtt{BK}_L || \mathtt{BK}_L)$$



[DSH25b] D. Shekhawat, *et al.* "Breaking the Quire Bottleneck: A Posit MAC Architecture with Reduced Adder Complexity." Conference on Next Generation Arithmetic, St. Louis, USA, 2025.

Results: ASIC Implementations and Comparison

Experimental Setup:

- Tool → Synopsys Design Compiler (T-2022.03-SP)
- Technology Node→ Nangate 45 nm and TSMC 28 nm

- The proposed work has an average improvement of
  - Area → 36.02%
  - Delay → 22.25%
  - Power → 26.68%

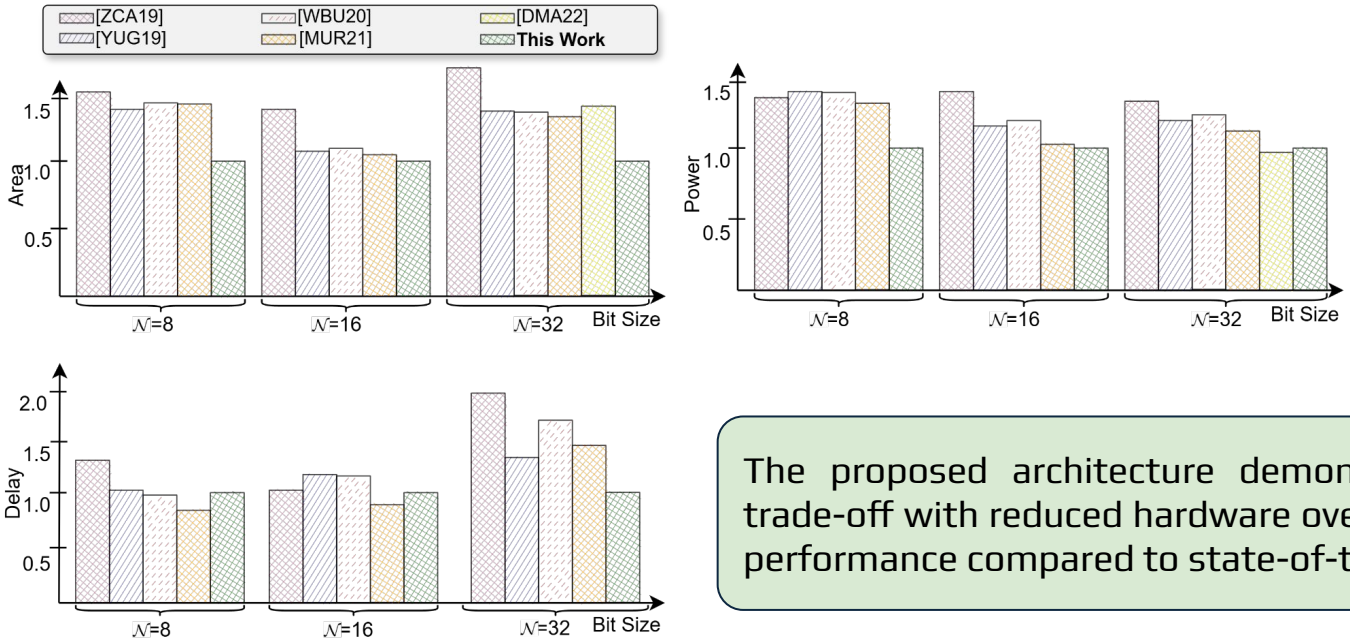
The power-performance–area (PPA) are improved using integrated regime-boundary detection with in-line shifting

ASIC Implementation and comparison with the state-of-the-art work

Work	*N*	Area (μm²)	Power (mW)	Delay (ns)
45 nm				
[ZCA19]	8	5349.45	4.46	4.38
	16	16147.42	12.18	11.70
	32	58937.59	40.07	42.01
[YUG19]	8	4234.78	4.78	3.16
	16	11011.83	11.88	6.09
	32	29045.79	33.28	6.00
[WBU20]	8	4590.87	4.73	3.02
	16	11263.02	12.37	5.99
	32	28366.06	34.79	11.94
[MUR21]	8	4555.25	4.18	2.66
	16	10710.77	10.26	4.57
	32	27267.44	30.51	7.15
[DMA22]	32	30419.12	26.07	—
This Work	**8**	**2617.83**	**2.88**	**3.09**
	16	**10211.43**	**10.04**	**5.08**
	32	**18315.32**	**26.86**	**4.09**
28 nm				
This Work	**8**	**1106.32**	**0.20**	**3.89**
	16	**2557.90**	**0.72**	**7.36**
	32	**4352.18**	**2.44**	**7.21**

[DSH25b] D. Shekhawat, *et al.* "Breaking the Quire Bottleneck: A Posit MAC Architecture with Reduced Adder Complexity," 2025 International CoNGA, Missouri, USA. [Presented]
[DMA22] D. Mallas´en, *et al.* "PERCIVAL: Open-source Posit RISC-V Core with Quire Capability," IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 3, pp. 1241–1252, 2022.
[WBU20] B. Wu, "SmallPositHDL." https://github.com/starbrilliance/SmallPositHDL, 2020. GitHub repository.
[YUG19] Y. Uguen, *et al.* "Evaluating the Hardware Cost of the Posit Number System," in 2019 29th International Conference on FPL, pp. 106–113, IEEE, 2019
[ZCA19] Z. Carmichael, *et al.* "Deep Positron: A Deep Neural Network using the Posit Number System," in 2019 DATE, pp. 1421–1426, IEEE, 2019.
[MUR21] R. Murillo, et al. "Energy-efficient MAC Units for Fused Posit Arithmetic," in 2021 IEEE 39th International Conference on Computer Design (ICCD), pp. 138–145, IEEE, 2021.

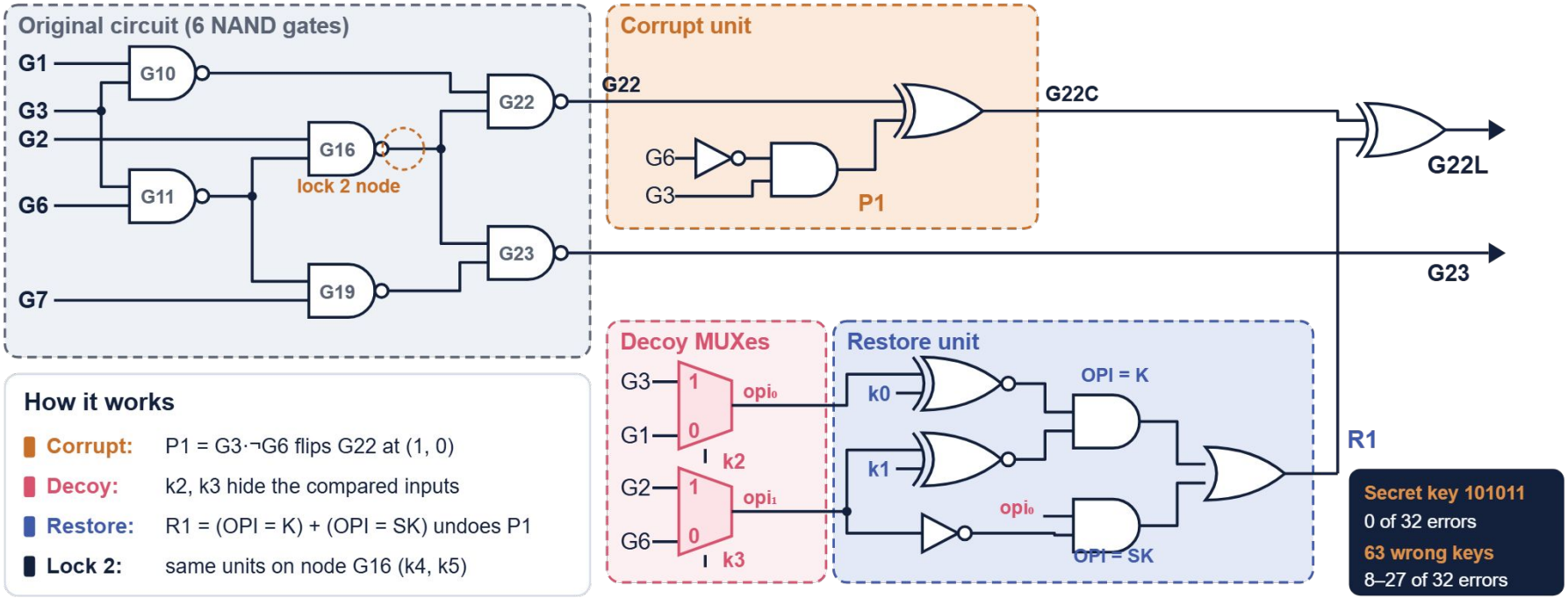
Results: Normalized APD Comparison

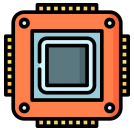


The proposed architecture demonstrates a balanced trade-off with reduced hardware overhead and improved performance compared to state-of-the-art designs

[DSH25b] D. Shekhawat, *et al.* "Breaking the Quire Bottleneck: A Posit MAC Architecture with Reduced Adder Complexity," 2025 International CoNGA, Missouri, USA. [Presented]
[DMA22] D. Mallas´en, *et al.* "PERCIVAL: Open-source Posit RISC-V Core with Quire Capability," IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 3, pp. 1241–1252, 2022.
[WBU20] B. Wu, "SmallPositHDL." https://github.com/starbrilliance/SmallPositHDL, 2020. GitHub repository.
[YUG19] Y. Uguen, *et al.* "Evaluating the Hardware Cost of the Posit Number System," in 2019 29th International Conference on FPL, pp. 106–113, IEEE, 2019
[ZCA19] Z. Carmichael, *et al.* "Deep Positron: A Deep Neural Network using the Posit Number System," in 2019 DATE, pp. 1421–1426, IEEE, 2019.
[MUR21] R. Murillo, et al. "Energy-efficient MAC Units for Fused Posit Arithmetic," in 2021 IEEE 39th International Conference on Computer Design (ICCD), pp. 138–145, IEEE, 2021.

Corrupt-and-Correct Obfuscation

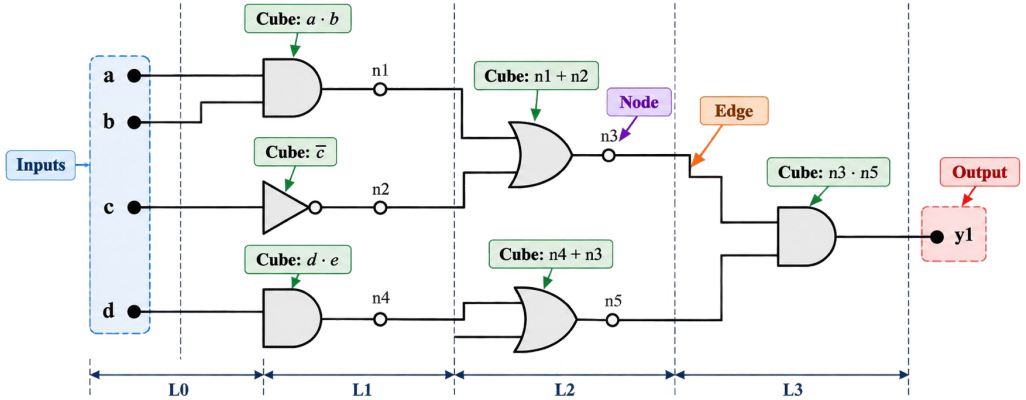




Experimental Setup

- Posit computing units were obfuscated using
  - AMD Xilinx Vivado 2022.1
- Converted to benchmark netlist (.bench) using custom Perl scripting
- IPs obfuscated using 16- and 32-bit keys to validate scalability
- Obfuscated benchmark (.bench) converted to Verilog(.v)
  - UC Berkeley ABC compiler
- To evaluate the hardware overhead, obfuscated netlists are synthesized using
  - Synopsys Design Compiler
  - TSMC 28 nm CMOS technology

Analysis of the Original Netlist



- Netlist represents a digital design structurally as
  - Logic gates, signal nodes, and interconnections from inputs to outputs

`add`: adder; `mult`: multiplier; `i2p`: integer-to-Posit; `p2i`: Posit-to-integer

Netlist	Inputs	Outputs	Nodes	Edges	Cubes	Levels
`64_add`	129	64	7696	14990	7945	391
`64_i2p`	33	64	915	1746	941	48
`64_p2i`	65	32	1278	2390	1320	43
`add`	65	32	3340	6462	3481	255
`i2p`	33	32	1006	1870	1036	65
`mult`	65	32	6619	12900	7749	244
`p2i`	33	32	1082	2034	1122	56

Overhead Analysis

Key	16-bit				32-bit			
Netlist	Nodes	Edges	Cubes	Levels	Nodes	Edges	Cubes	Levels
64_add	7752	15095	8019	391	7810	15201	8093	391
64_i2p	971	1851	1015	48	1029	1957	1089	48
64_p2i	1334	2495	1394	43	1392	2601	1468	43
add	3396	6567	3555	255	3454	6673	3629	255
i2p	1062	1975	1110	65	1120	2081	1184	65
mult	6675	13005	7823	244	6733	13111	7897	244
p2i	1138	2139	1196	56	1196	2245	1270	56

- Increasing datapath width from 16-bit to 32-bit
  - Adds 58 nodes, 106 edges, and 74 cubes per netlist,
  - Identical logic depth

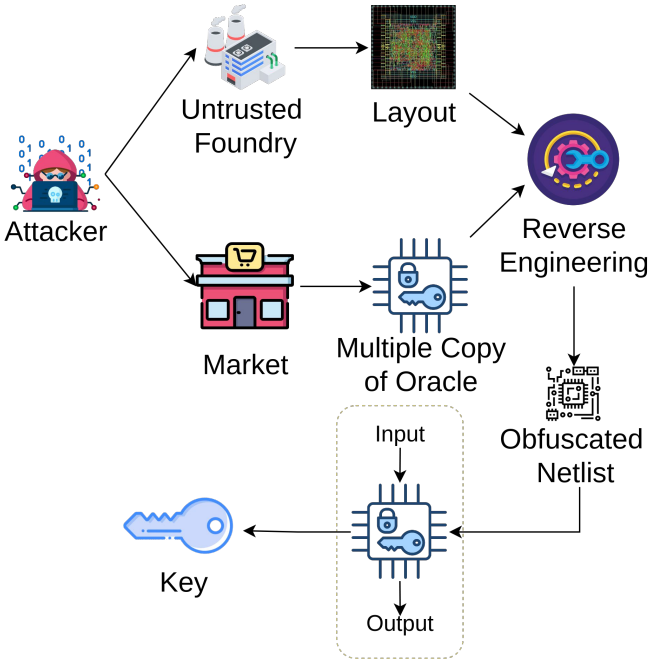
Overhead Analysis...

Netlist	Key	Area		Power		Delay	
		μm^2	%	μW	%	ns	%
64_add	16	1,503.03	1.03	119.76	-0.49	7.97	-0.25
64_i2p		173.56	9.05	5.79	10.23	2.15	5.91
64_p2i		233.63	7.19	10.32	7.62	2.39	0.42
add		692.57	4.14	51.25	2.37	5.15	-0.77
i2p		193.94	9.94	6.22	7.09	2.46	-3.53
mult		1,882.68	1.12	225.90	0.08	5.30	-2.39
p2i		213.84	6.54	10.12	2.47	2.33	-15.58
64_add	32	1,522.63	2.35	123.15	2.33	7.99	0.00
64_i2p		187.77	17.98	6.00	14.11	2.15	5.91
64_p2i		251.76	15.51	10.65	11.05	2.43	2.10
add		704.91	6.00	51.50	2.87	5.19	0.00
i2p		207.17	17.44	6.61	13.75	2.52	-1.18
mult		1,903.94	2.26	226.01	0.12	5.34	-1.66
p2i		231.38	15.28	10.45	5.77	2.38	-13.77

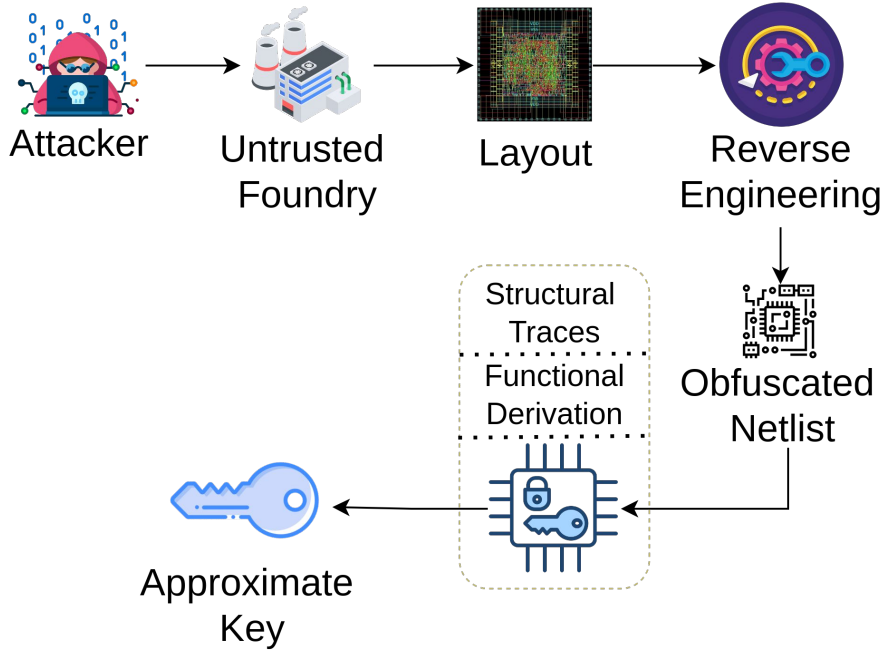
- Overhead for **16-bit**
  - Area: 5.58 %
  - Power: 4.19 %
  - Delay: -2.31 %
- Overhead for **32-bit**
  - Area: 10.98 %
  - Power: 7.14 %
  - Delay: -1.23 %
- Low overhead for hardware security integration

Key-recovery Attack-based Security Analysis

Oracle-guided (OG) Attacks

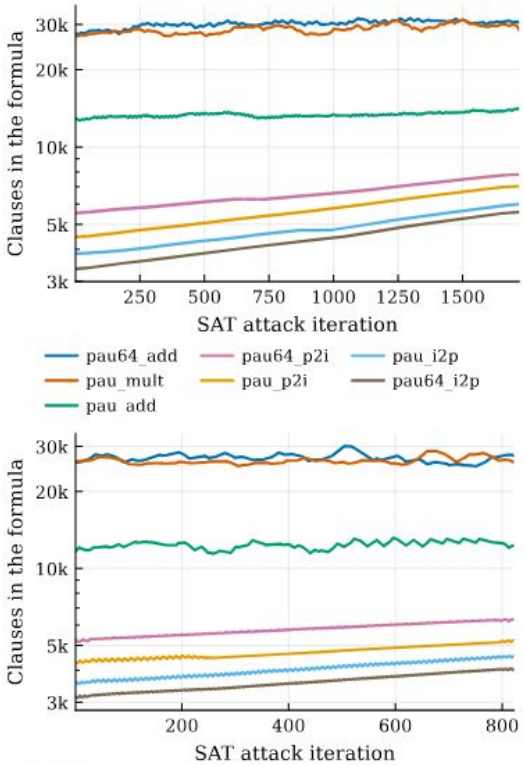


Oracle-less (OL) Attacks



SAT Attack Evaluation Results

Netlist	Key	Iteration	Variables	Clauses	Decisions	RT(s)	MRSS
64_add	16	4097	62985590	33137	2699413	OoT	1323.51
64_i2p		11279	20418870	22582	581769	OoT	517.95
64_p2i		9289	24156796	22251	782077	OoT	552.73
add		6178	41542156	20024	2828032	OoT	1017.54
i2p		10135	20821562	22235	538195	OoT	518.12
mult		822	10921492	25292	11700562	OoT	439.64
p2i		9870	21777796	22046	515333	OoT	537.90
64_add	32	4033	62470258	39160	2272893	OoT	1186.98
64_i2p		10031	19323850	19250	680656	OoT	551.22
64_p2i		8435	22915120	21132	866689	OoT	567.02
add		5677	38833230	22312	2027354	OoT	991.68
i2p		9089	19727666	19331	615758	OoT	545.68
mult		1719	23009998	27200	7969433	OoT	616.58
p2i		8910	20693860	21243	608967	OoT	538.03



- SAT attack resiliency with exponential key recovery complexity

SCOPE Attack Evaluation Results

Netlist	16-bit					32-bit				
	KPA1	KPA2	MKPA	RT(s)	MRSS	KPA1	KPA2	MKPA	RT(s)	MRSS
64_add	0.00	0.00	0.00	3.37	31.29	0.00	0.00	0.00	5.98	31.38
64_i2p	0.00	0.00	0.00	1.33	27.61	0.00	0.00	0.00	2.06	27.66
64_p2i	0.00	0.00	0.00	1.18	27.72	0.00	0.00	0.00	2.20	27.72
add	0.00	0.00	0.00	1.86	28.72	0.00	0.00	0.00	3.84	28.75
i2p	0.00	0.00	0.00	1.09	27.54	0.00	0.00	0.00	2.05	27.68
mult	0.00	0.00	0.00	3.36	30.76	0.00	0.00	0.00	6.31	30.82
p2i	0.00	0.00	0.00	0.99	27.74	0.00	0.00	0.00	2.41	27.80

- No correct key recovery for either key configuration

Low Overhead High Attack Resiliency → Balanced Tradeoff



Conclusion

- Arithmetic IP faces hardware security challenges
  - No prior obfuscation scheme targets Posit arithmetic units.
- Solution: Design-time countermeasure using logic locking
- Proposed Corrupt-and-correct obfuscation that protects Posit I2P, P2I, adder and multiplier units
  - Perturb unit corrupts protected patterns; only correct key restores them
- Results :
  - 16/32-bit key obfuscated IPs shows 5.58/10.98% area, 4.19/7.14% power , −2.31/−1.23% delay overhead on TSMC 28 nm CMOS technology
  - SAT attacks time out; SCOPE reports 0.00% KPA
- Low-overhead protection for Posit units for scientific accelerators



References

[1] S. Vishwakarma, *et al.*, "Impact of MAC Unit Design Architectures and Their Applications in Modern Computing: An In-Depth Review," IEEE Access, 2026.

[2] V. Mishra, M. Rathor, and U. Chatterjee, "Dual-Mode Rounding Algorithms and Hardware for Posit-Based DNN Training: The Future of Mixed Precision Frameworks," ACM Transactions on Embedded Computing Systems, vol. 25, no. 1, pp. 1–26, 2026.

[3] J. Gandhi, D. Shekhawat, M. Santosh, J. Dofe, and J. G. Pandey, "Functional Obfuscation of Lightweight Crypto Cores for Improved Security against IP Piracy," The Journal of Supercomputing, vol. 82, no. 13, p. 669, 2026.

[4] K. Z. Azar, H. M. Kamali, F. Farahmandi, and M. Tehranipoor, Understanding Logic Locking. Springer, 2024.

[5] K. Raj, *et al.*, "SENTRY: Protecting System-on-Chip Designs against Supply-Chain Attacks," ACM Trans. Embed. Comput. Syst., vol. 25, Apr. 2026.

[6] P. B. Roy, A. Saha, J. Knechtel, M. Nabeel, S. Sreekumar, L. Mankali, D. Mukhopadhyay, R. Karri, and O. Sinanoglu, "NiLoPher: Breaking a Modern SAT-Hardened Logic-Locking Scheme via Power Analysis Attack," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, pp. 1–1, 2026.

[7] M. Heidary and B. K. Joardar, "Attacking AI Accelerators via Arithmetic Properties," Journal of Hardware and Systems Security, vol. 10, no. 1, p. 19, 2026.

[8] Posit Working Group, "Standard for Posit Arithmetic (2022)," 2022. https://posithub.org/docs/posit_standard-2.pdf.

[9] J. A. Roy, F. Koushanfar, and I. L. Markov, "Methods for Protecting Against Piracy of Integrated Circuits," Nov. 11 2010. US Patent App. 12/720,634.

[10] Y. Aghamohammadi, H. Jin, and A. Rezaei, "GALA: An Explainable GNN-based Approach for Enhancing Oracle-Less Logic Locking Attacks Using Functional and Behavioral Features," in 2026 31st Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 924–930, IEEE, 2026.

[11] L. Li and A. Orailoglu, "Residual Structural Traces in Logic Locking: Insights from Redundancy Analysis and Beyond," IEEE Design & Test, 2026.

[12] J. Gandhi, N. Handa, A. Nayak, D. Shekhawat, *et al.*, "SHAKTI: Securing Hardware IPs by Cascade-gated Multiplexer-based Logic Obfuscation," in 2025 38th International Conference on VLSI Design and 2024 23rd International Conference on Embedded Systems (VLSID), pp. 139–144, IEEE, 2025.

[13] L. Aksoy, M. Yasin, and S. Pagliarini, "CAC 2.0: A Corrupt and Correct Logic Locking Technique Resilient to Structural Analysis Attacks," in 2024 IEEE 25th Latin American Test Symposium (LATS), pp. 1–6, IEEE, 2024.

[14] D. Shekhawat, M. Pangoria, V. Sreejith, J. Gandhi, J. G. Pandey, and C. Shekhar, "Parameterized Low-Overhead Data Conversion Architectures for Posit Arithmetic," in 2025 IEEE 68th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 69–73, IEEE, 2025.

[15] D. Shekhawat, J. Gandhi, M. Santosh, and J. G. Pandey, "PHAc: Posit Hardware Accelerator for Efficient Arithmetic Logic Operations," in Conference on Next Generation Arithmetic, pp. 88–100, Springer, 2023



Thank You !

For collaboration or queries contact:
singhdiksha281@gmail.com
diksha.shekhawat@csir.res.in
diksha.ceeri22j@acsir.res.in