The 26th International Symposium on Quality Electronic Design (ISQED'25)

REDACTOR: eFPGA Redaction for DNN Accelerator Security

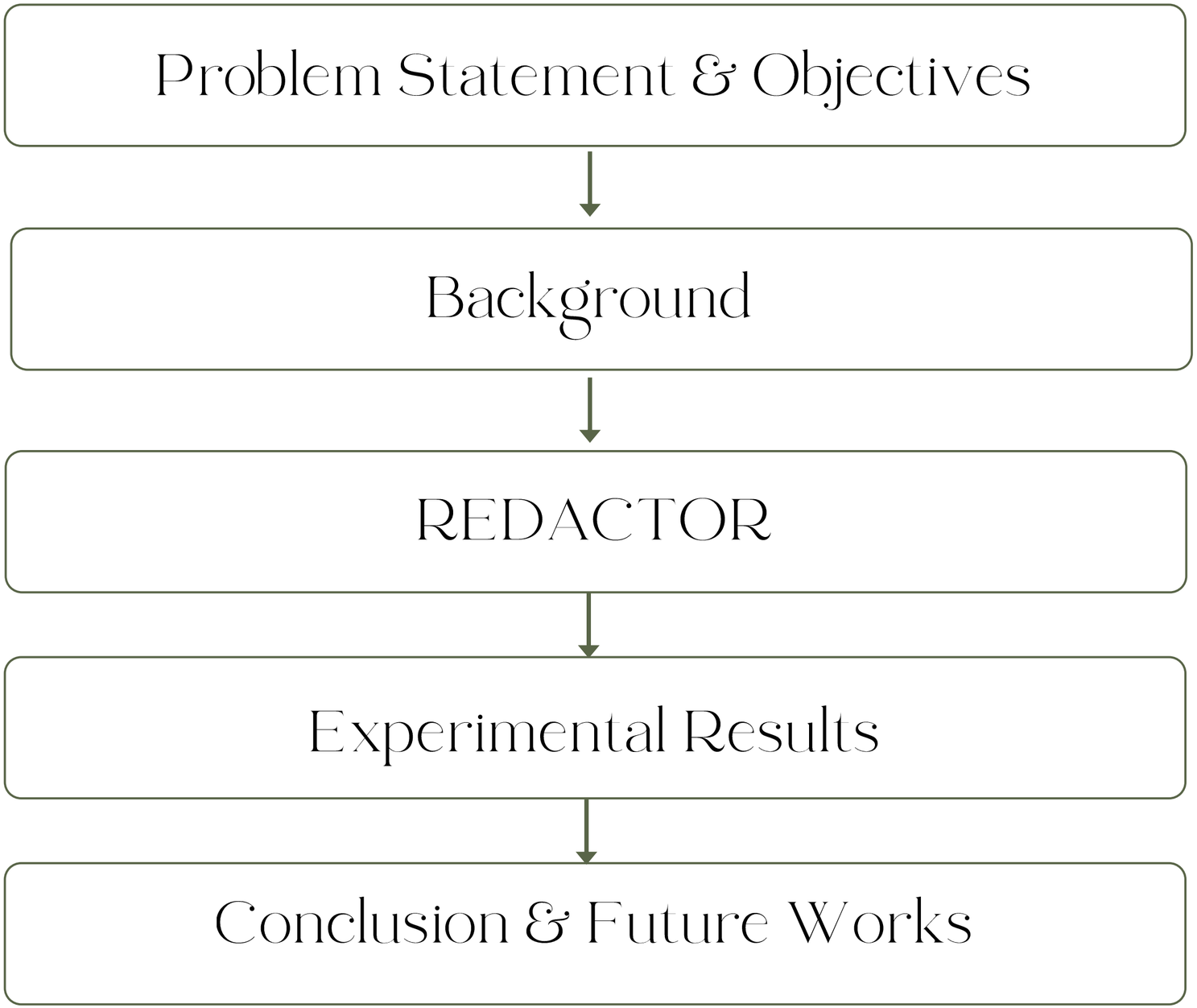
Yazan Baddour, Ava Hedayatipour, and Amin Rezaei

ISQED Symposium 2025

Speaker Bio

Dr. Amin Rezaei is an Assistant Professor in the Department of Computer Engineering and Computer Science at California State University, Long Beach. He obtained his Ph.D. in Computer Engineering from Northwestern University. He has a decade of experience in hardware security, computer architecture, and machine learning, with more than 50 peer-reviewed scientific articles at flagship venues such as DAC, ICCAD, DATE, and ASP-DAC. He is a senior member of IEEE and a lifetime member of ACM and AAAI and has served on the technical program committees of many major conferences in his area.

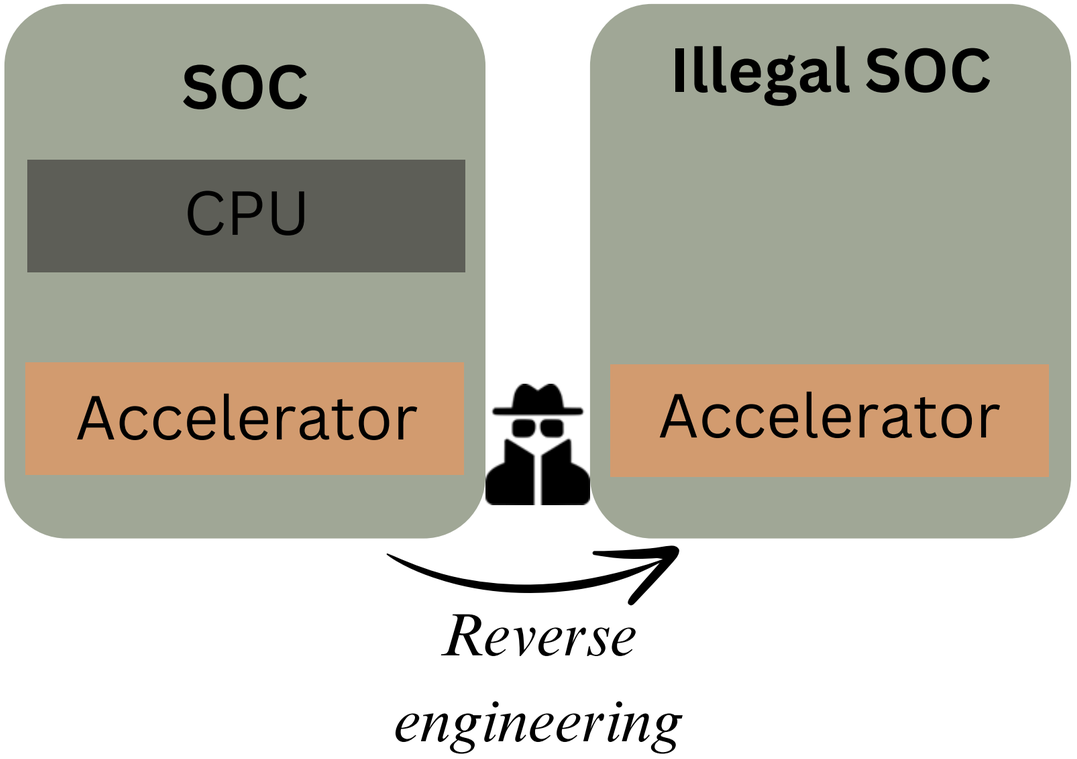
Agenda


Problem Statement & Objectives
Background
REDACTOR
Experimental Results
Conclusion & Future Works

Problem Statement & Objectives

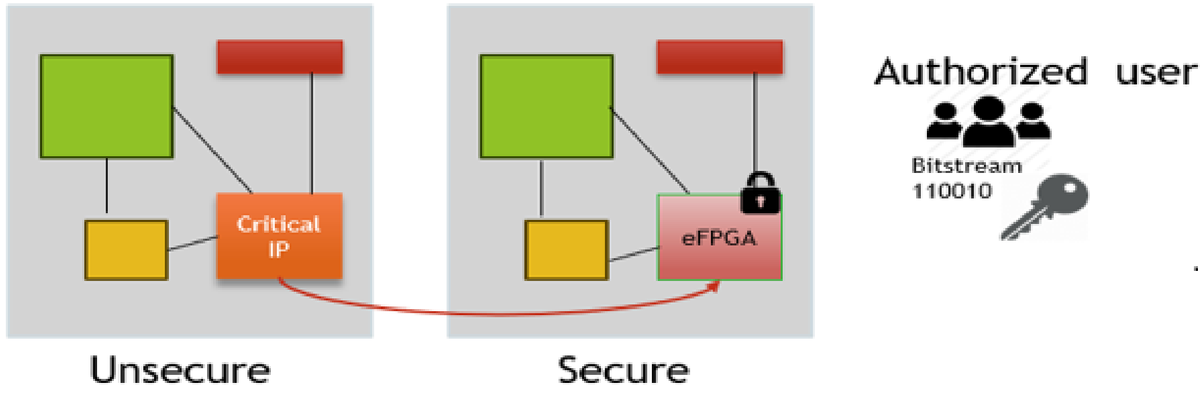
Problem Statement

- Developing hardware accelerators often involve third-party silicon fabs.
- Relying on third-party silicon fabs reduces the security control of hardware accelerators.
- Accelerators are being used in critical tasks such as autonomous vehicles and medical care.
- Existing logic locking methods to secure ICs are vulnerable to oracle-guided attacks like SAT.



Objectives

- Introduce an eFPGA-based redaction flow to hide different parts of an accelerator (RTL to GDSII).
- Give informed decision on how to choose a critical IP.
- Try to pay a minimum cost of overheads like power delay and area.
- Evaluate the security against oracle-guided SAT attacks.

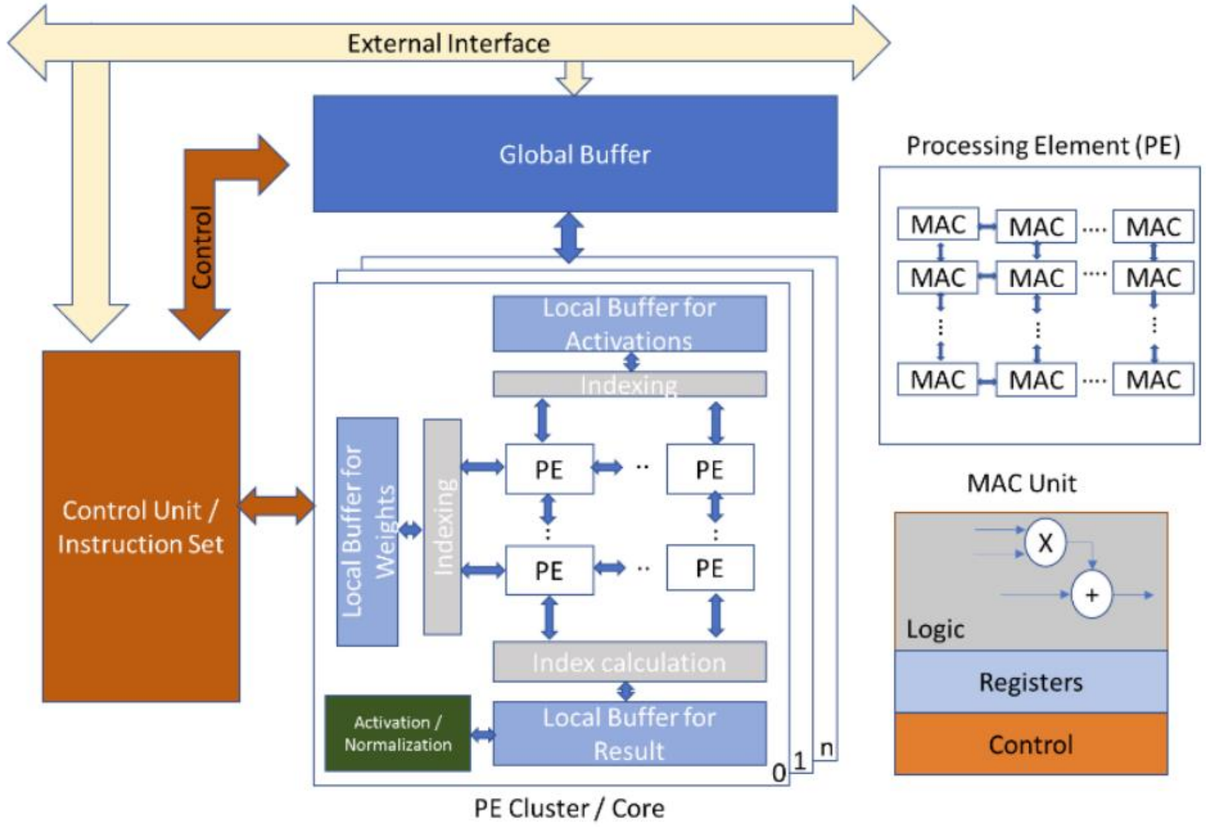


Background

Deep Neural Networks

- Some DNNs require over a billion Multiply And Accumulate (MAC) operations.
- Each operation needs four memory access.
- General CPUs cannot handle such computing loads.

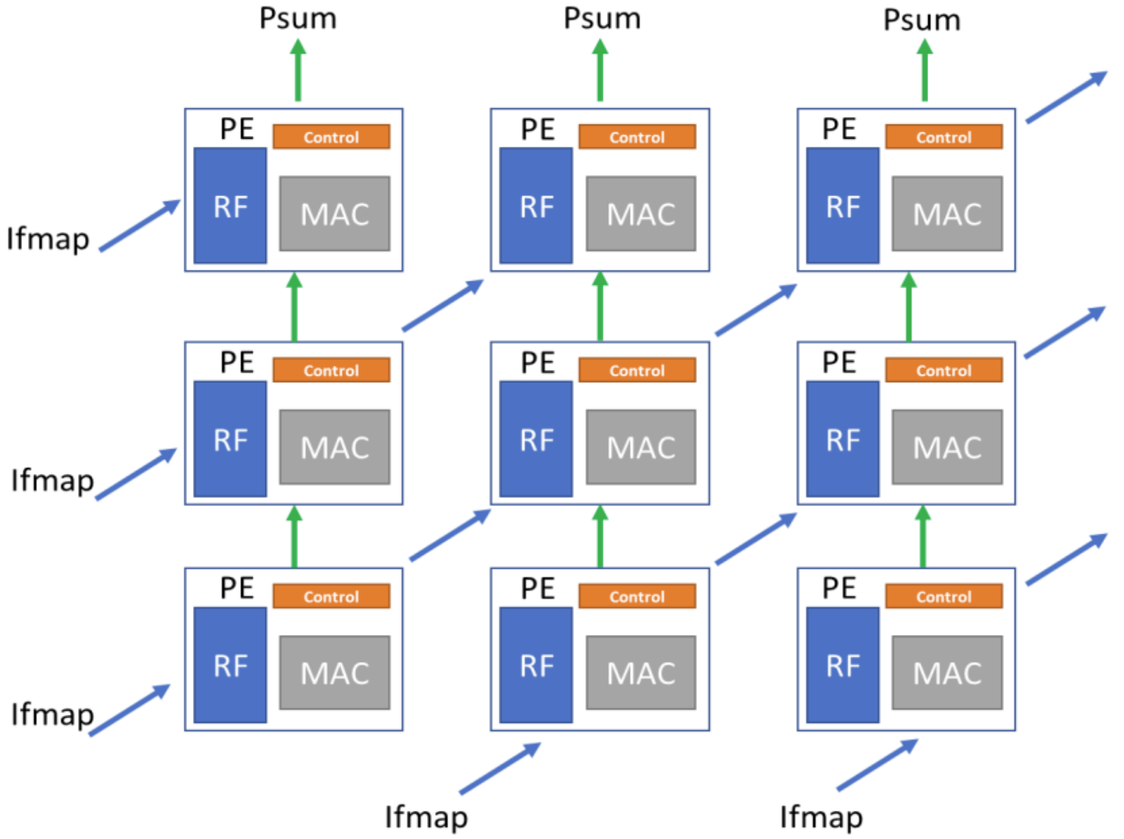
WE NEED A SPECIFIC HARDWARE FOR THAT!



Raju Machupalli, “Hardware Accelerators for Deep Neural Networks, Master’s Thesis,” Department of Electrical and Computer Engineering University of Alberta, 2023.

Hardware Accelerators

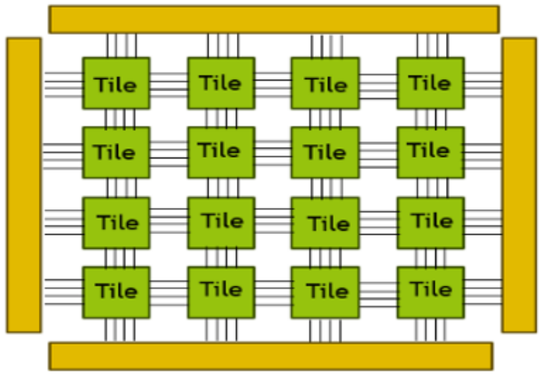
- Provides superior efficiency and performance for DNN applications.
- **Data flow accelerators:** Reduce energy movement of operands (weight, input, partial sums). → We utilize this type.
- **ALU based accelerators:** Modify the MAC unit to have large computing resources.
- **Sparse based accelerators:** Compress non-zero weights and skip zero multiplication to reduce computational and memory requirement.



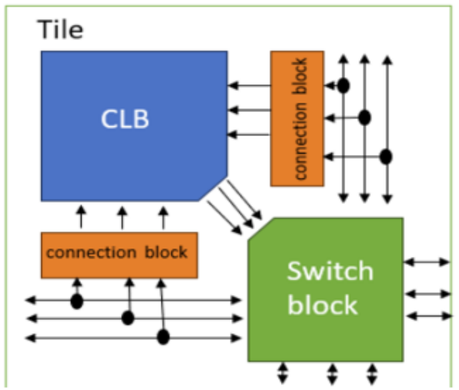
Y.-H. Chen et al., "Eyeriss v2: A Flexible Accelerator for Emerging Deep Neural Networks on Mobile Devices," IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 9, no. 2, pp. 292–308, 2019.

Field Programmable Gate Arrays

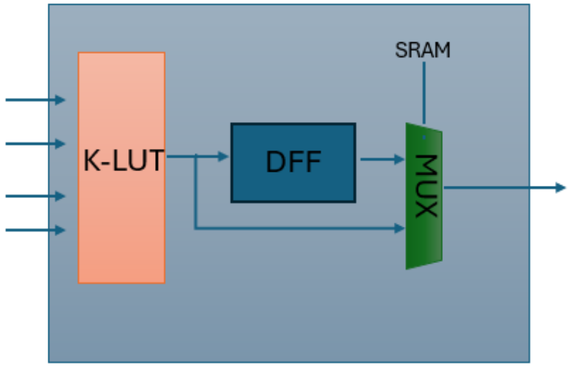
- FPGA provides flexibility through reconfiguration post manufacturing.
- Modern FPGAs adopt a tile-based structure and I/O cells surrounding.
- Each tile has Configurable Logic Block (CLB) and programable routing.
- Each CLB has *N* Basic Logic Element (BLE).
- Each BLE has a *K* input lookup table, flipflop and 2x1 multiplexer.



a) 4x4 fabric



b) Tile components



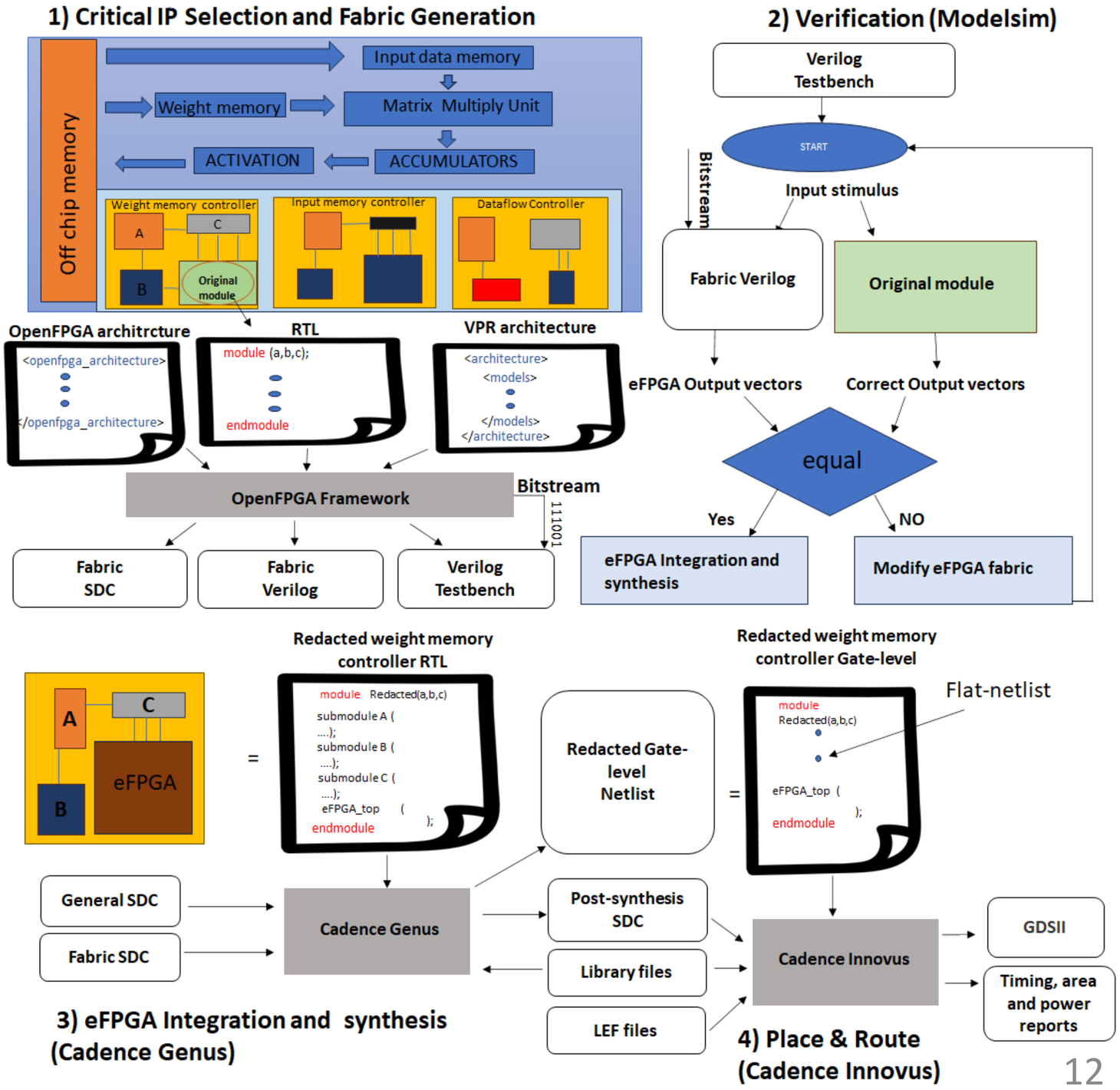
c) BLE components

REDACTOR

- We propose REDACTOR: an eFPGA REDaction framework for CNN ACcelera TORs.

- We utilize the CNN accelerator outlined in [*], which is a modification of the dataflow accelerator initially proposed in [%].

Note: The proposed eFPGA redaction flow is independent of the chosen accelerator.



[*] https://github.com/8krisv/CNN-ACCELERATOR
[%] J. Jo et al., "Energy-Efficient Convolution Architecture Based on Rescheduled Dataflow," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 65, no. 12, pp. 4196-4207, 2018,

Critical IP Selection

- Opting for a module that significantly impacts the outputs → Beneficial for causing output corruption when an unauthorized user loads the wrong bitstream.

- Selecting a module that causes error propagation throughout the system → Amplify the impact of using incorrect bitstreams and decrease the accuracy of the DNN model.

- Choosing a module that can be implemented using a complex fabric with a large unroll factor → beneficial for securing against existing attacks

Critical IP	# of Modules	# of I/Os	Description
OWMC	5	20	Controls the flow of DNN weights
MUXDC	4	15	Controls dataflow between PEs
OMDC	20	26	Controls the convolution process
PEDC	1	5	Sets/resets output register of PEs

eFPGA Fabric Generation

- Utilize OpenFPGA [*].

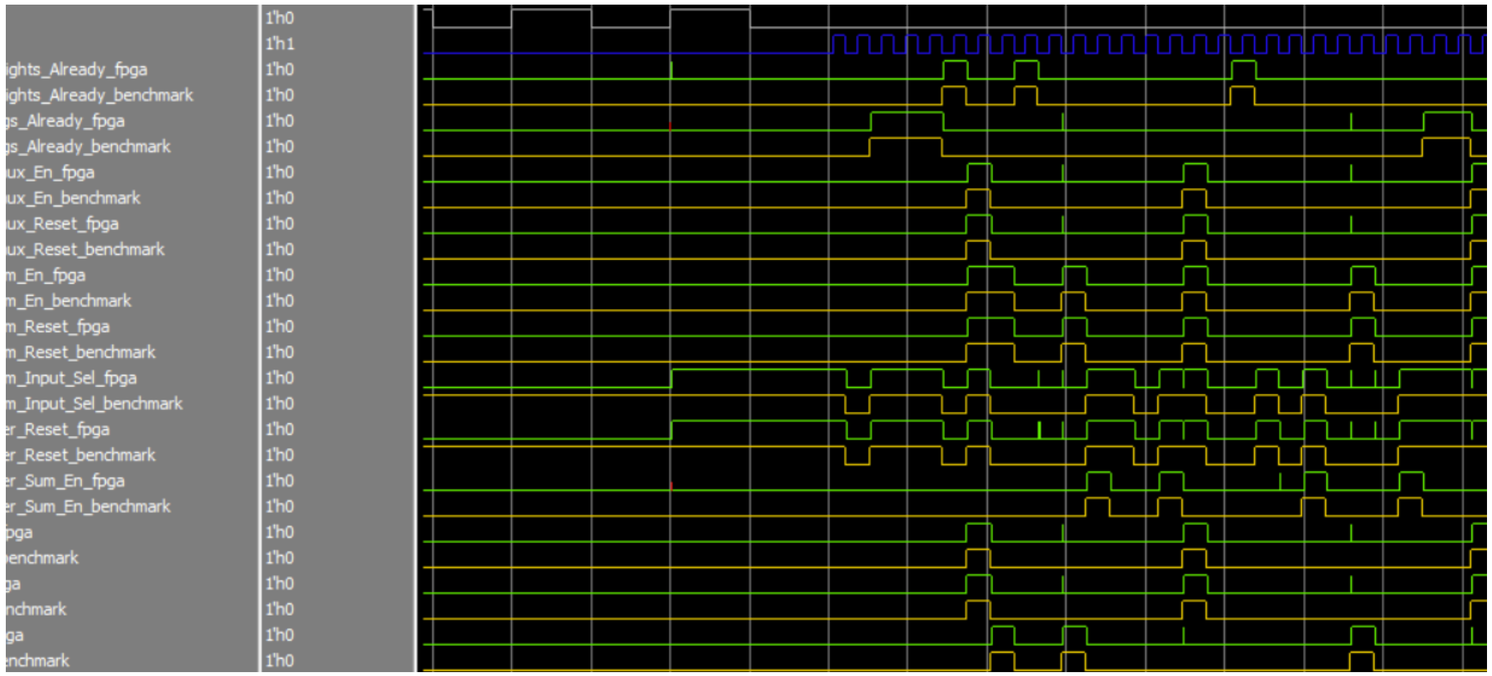
- We select a fabric architecture to achieve 100% block utilization and over 90% I/O utilization → Manually adjusting the number of BLEs (N) per CLB and the number of I/O pins per tile until the target utilization is reached.

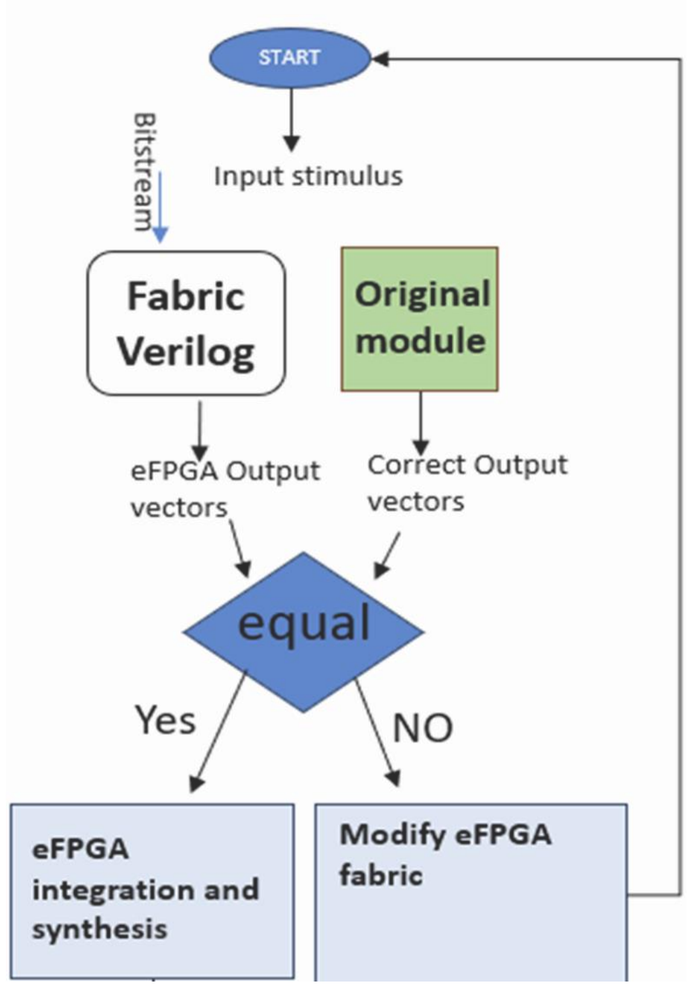
Parameter	Value	Description
K	4	Input size of a LUT/FLUT
N	[1,9]	Number of BLEs per CLB
W	Auto	Number of routing tracks in a channel
Fc_{in}	0.15	Fraction of the routing tracks that a CLB input can connect
Fc_{out}	0.1	Fraction of the routing tracks that a CLB output can connect
Fs	3	Number of connections per incoming routing track in a switch block
L	4	The length of routing track (number of CLBs spanned)

[*] Xifan Tang et al. “OpenFPGA: An Open-Source Framework for Agile Prototyping Customizable FPGAs,” IEEE Micro, vol. 40, no. 4, pp. 41-48, 2020.

Verification

- Utilize ModelSim.



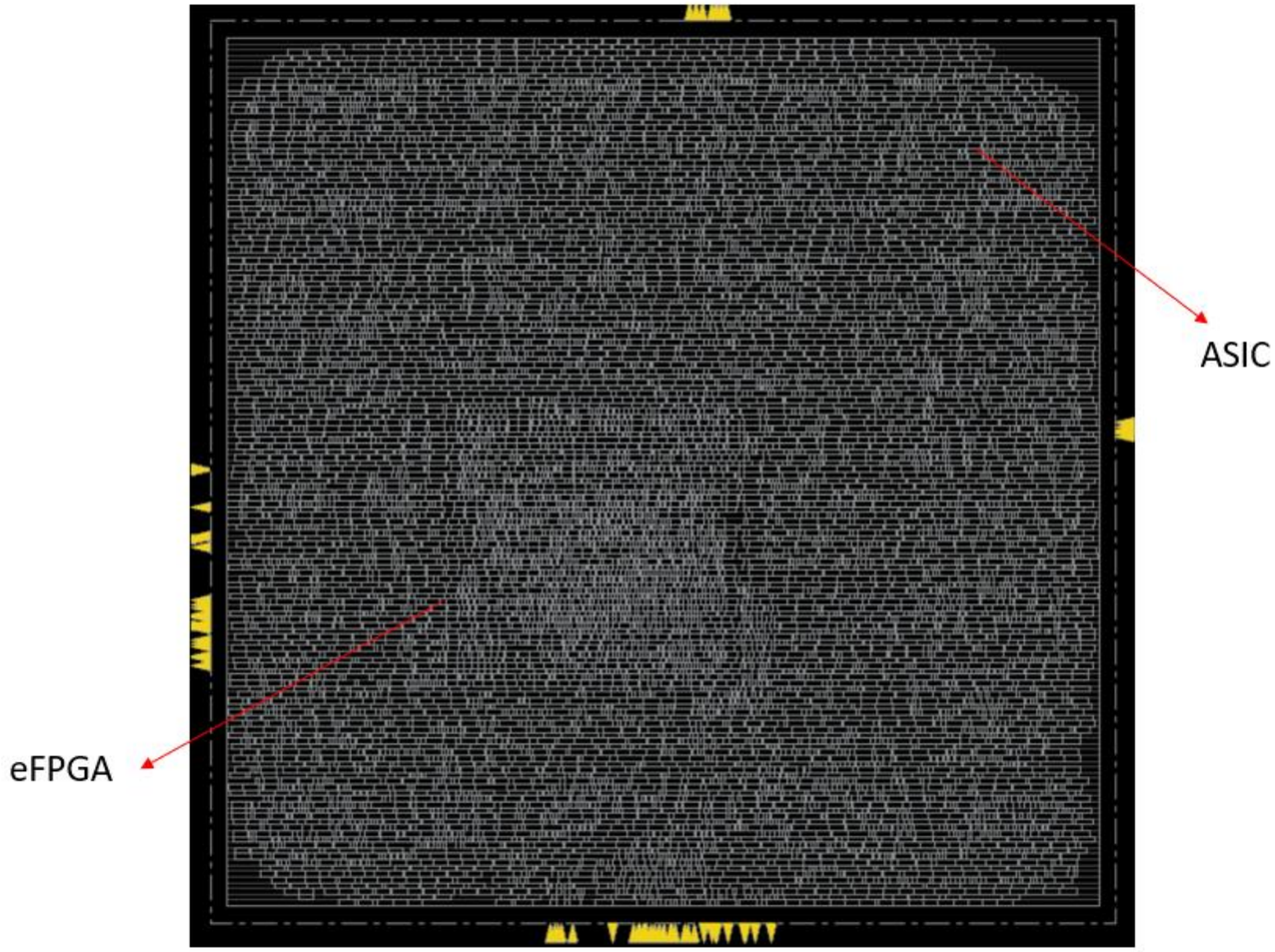


eFPGA Integration and Synthesis

- Utilize Cadence Genus.
- 45nm technology.
- We choose to use a flattened netlist to simplify the place and route stage.

Place & Route

- Utilize Cadence Innovus
- We choose to use a flattened netlist to simplify the place and route stage.



Final optimized layout

Experimental Results

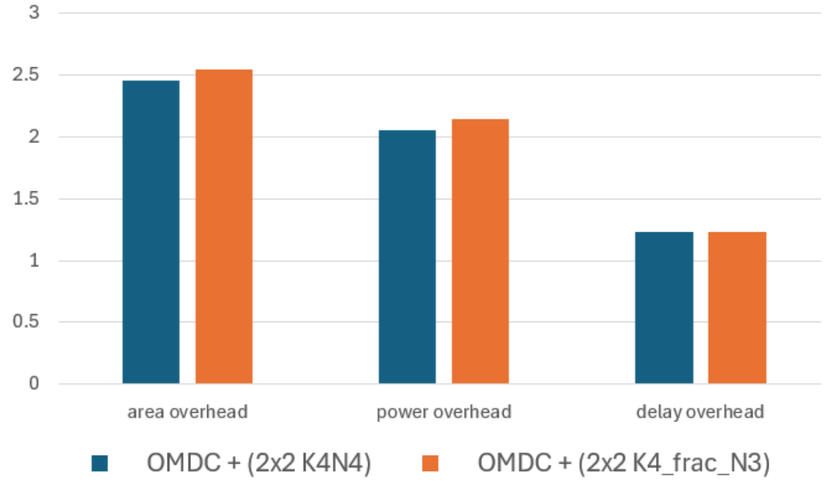
eFPGA fabric characteristics

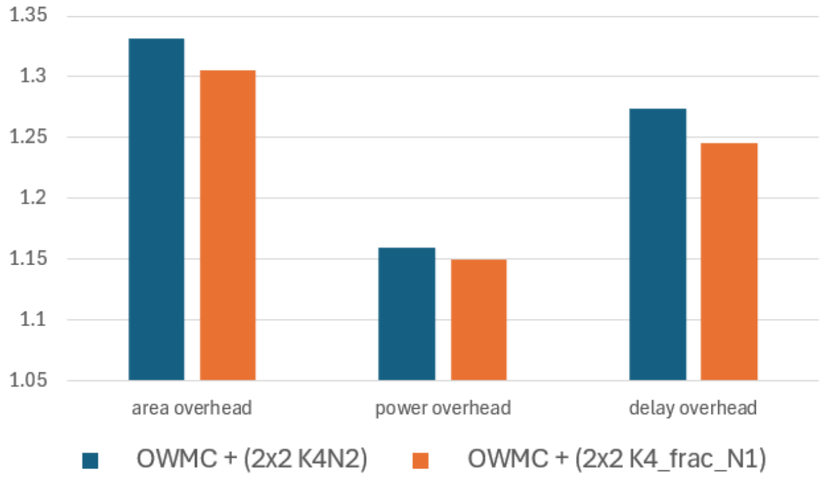
eFPGA Fabric	Block Utilization	I/O Utilization	Bitstream size	Channel width
2x2 K4N2	100%	94%	614	18
2x2 K4 frac N1	100%	94%	458	18
1x1 K4N6	100%	100%	440	26
1x1 K4 frac N3	100%	100%	256	18
2x2 K4N4	100%	95%	1160	30
2x2 K4 frac N3	100%	95%	1059	30
1x1 K4N1	100%	100%	66	6
1x1 K4 frac N1	100%	100%	79	14

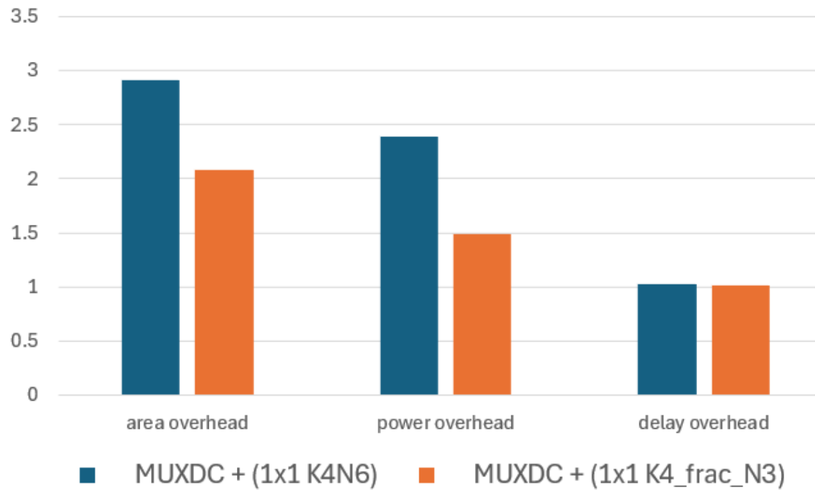
Security results

eFPGA Fabric	Unroll	# Clauses	Time (s)	Key Reported?
2x2 K4N2	59	1610318	99	YES
2x2 K4 frac N1	30	559536	7	YES
1x1 K4N6	36	875500	51	YES
1x1 K4 frac N3	2	9291	0.26	YES
2x2 K4N4	112	N/A	Time out	NO
2x2 K4 frac N3	59	3299375	81	YES
1x1 K4N1	5	4189	0.1	YES
1x1 K4 frac N1	2	2384	0.08	YES

Overhead Analysis







OMDC OWMC MUXDC

- For the OMDC IP, choosing a regular LUT is preferable.
- Opting for FLUT-based is preferable when integrating eFPGA for the OWMC IP.
- For the MUXDC IP, opting for FLUT proves preferable for redaction.
- In the case of PEDC IP, choosing the fabric with regular LUTs is preferable. However, it is not efficient to replace a small IP with eFPGA due to the overheads introduced.

Conclusion & Future works

- Redacting with fully utilized eFPGA fabrics introduced a delay overhead up to 45%.
- FLUTs reduce the number BLEs or CLBs, leading to lower power consumption, reduced delay, and minimal area overhead.
- While FLUT-based eFPGAs enhance efficiency, they exhibit lower unroll factors, making it easier for adversaries to extract the bitstream.
- Increasing the unrolling factor in eFPGA fabrics systematically to make attacks more difficult.
- Introducing non-unrollable cycles to enhance security while ensuring that power and area overheads remain low.

Yazan Baddour

Ava Hedayatipour

Amin Rezaei

THANK YOU

https://github.com/cars-lab-repo/REDACTOR