DEEP: Developing Extremely Efficient Runtime On-Chip Power Meters

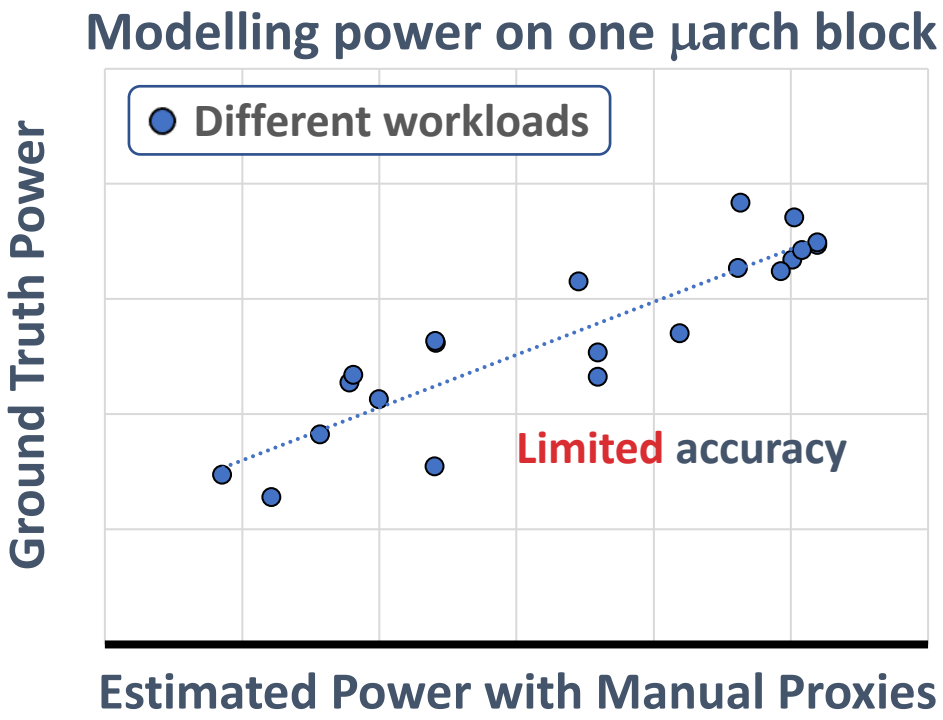
Zhiyao Xie[1], Shiyu Li[2], Mingyuan Ma[2], Chen-Chia Chang[2], Jingyu Pan[2], Yiran Chen[2], Jiang Hu[3]

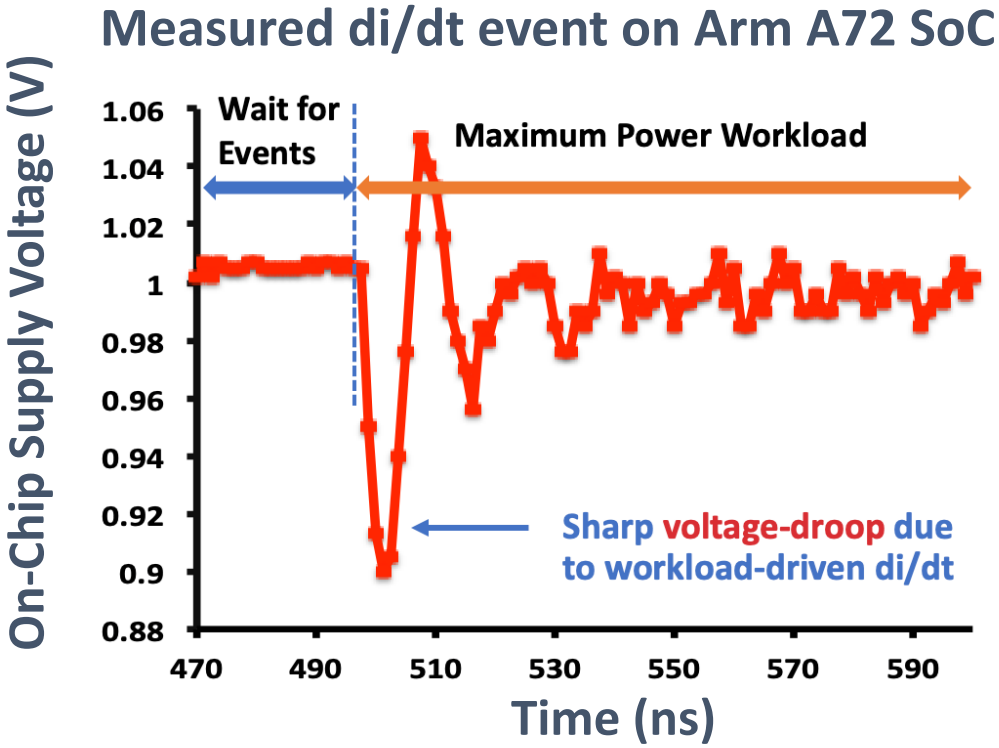
[1]Hong Kong University of Science and Technology,

[2]Duke University, [3]Texas A&M University

IEEE/ACM ICCAD 2022

Background: Difficulty in Runtime Power Modeling



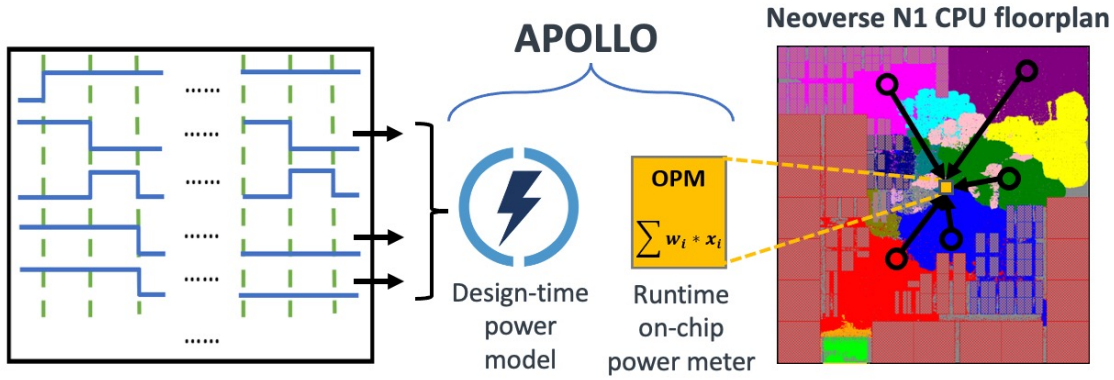


- **Peak-Power mitigation requires accurate power estimation to drive throttling**
  - Manually inferring proxies is very difficult in complex modern CPUs
- **Abrupt changes in CPU current-demand (di/dt event) leading to deep voltage-droop**

Background: A Recent Prior Work Named APOLLO*

Runtime Challenges Summary

- Peak power mitigation
  - **Difficult to manually** infer proxies
- Voltage droop (Ldi/dt) mitigation
  - Require very **low** response latency
- On-chip power modeling
  - **High** overhead to implement on HW
  - **Limited** temporal-resolution



- **Automated** development of on-chip power model (OPM) as part of hardware
- Selects power-related signals (proxies) as OPM input
- **Single-cycle** temporal resolution in the OPM

*Zhiyao Xie, et al., APOLLO [MICRO'21] (Best Paper)

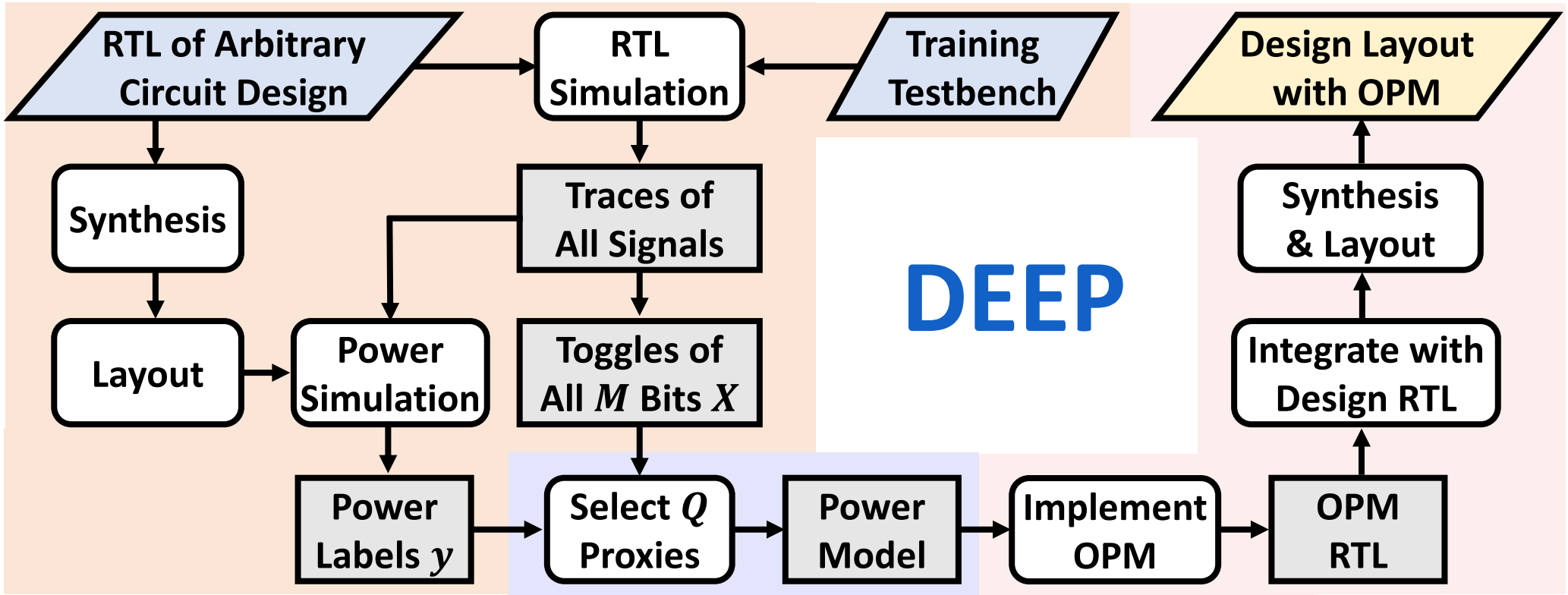
Background: Comparison with Prior Works

Baseline Methods	Model Input Candidate V_M (Candidate Count M)	Input Selection Method	Power Estimation Level	Temporal Resolution	Claimed OPM Area Overhead
B1. MICRO'21 [38]	All RTL signals (178 K)	MCP	Design-level	Per-cycle	< 1%
B2. MICRO'19 [20]	All RTL signals (178 K)	K-means	Design-level	100s cycles	N/A
B3. DATE'18 [25]	Registers (67 K)	Lasso	Design-level	> 1K cycles	7%
B4. DATE'18 [41]	Module I/O signals (< 178K)	Increase by level	Component-level	100s cycles	4 – 10%
B5. ASPDAC'15 [39]	Registers (67 K)	No Selection	Design-level	Per-cycle	16%
DEEP (this work)	All bits of RTL signals (578 K)	Two-step Selection	Component-level	Per-cycle	**< 0.1%**

Overview of representative works in proxy-based on-chip power estimation

- Achieve much **higher** efficiency with similar accuracy:
  - Support a larger number of candidates
  - Adopt a new two-step signal selection method
- Support **component-level** on-chip power model
- Keeps **per-cycle** resolution + **fully-automated** model development

Method: The DEEP OPM development framework

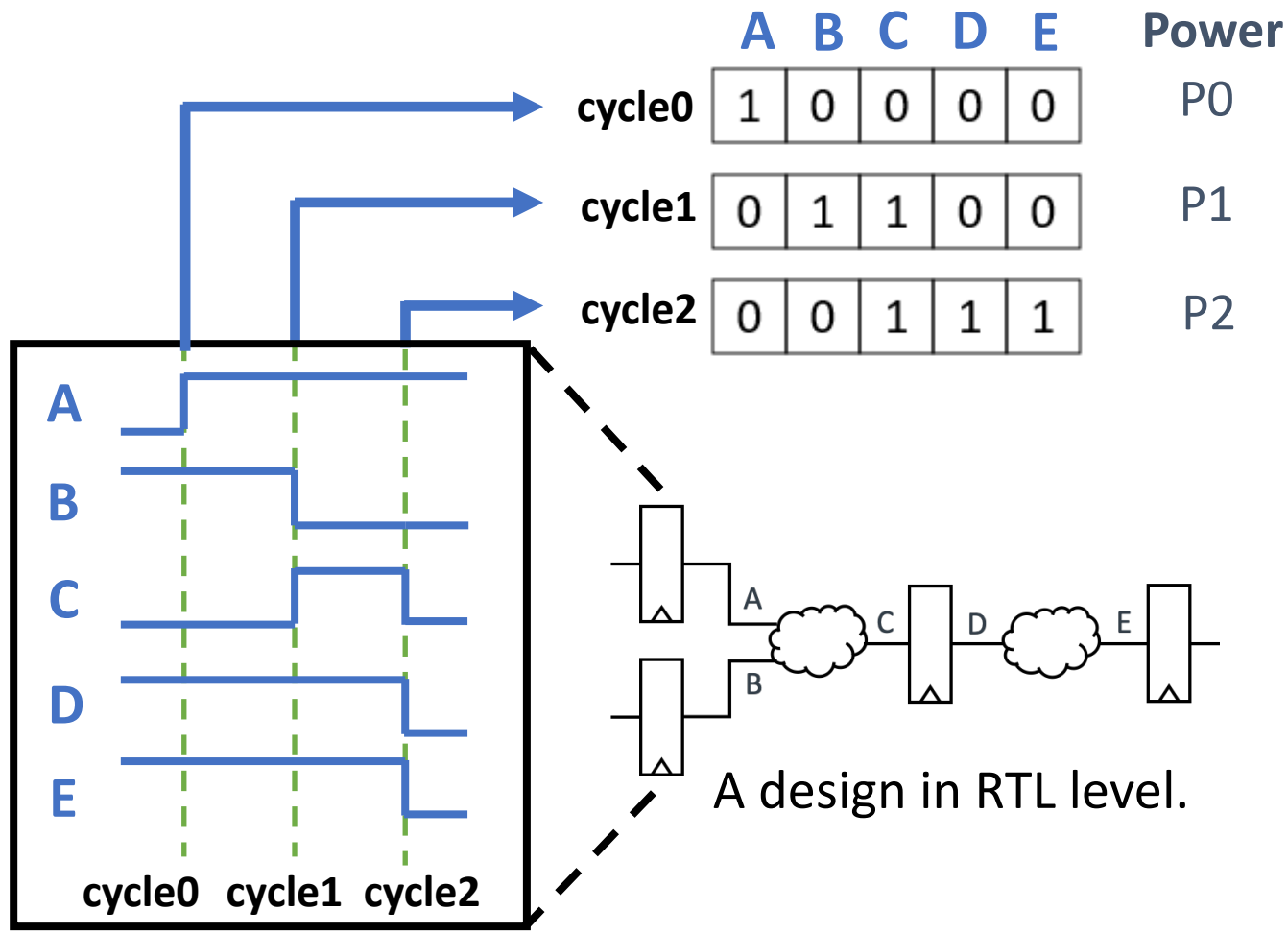


Part 1: Generate dataset with **togging/waveform of all signals** and simulated **power labels**

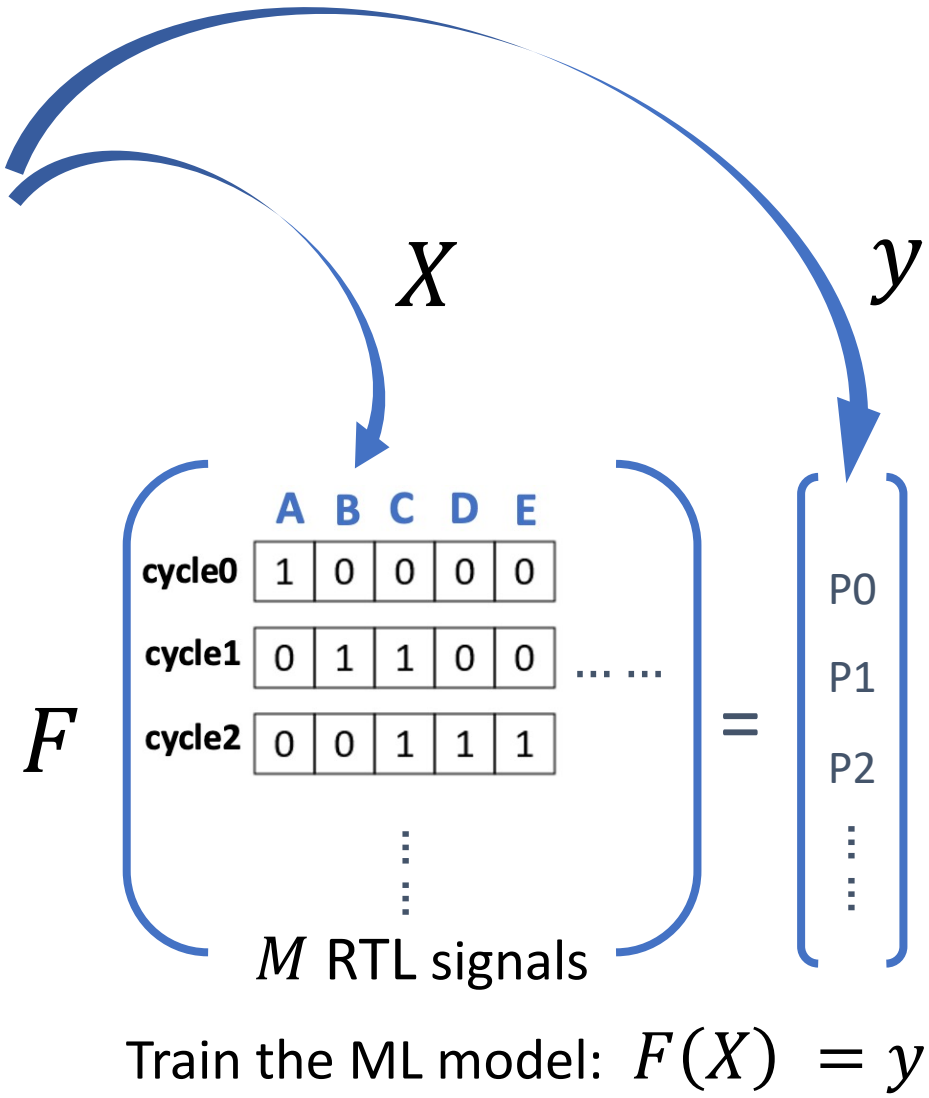
Part 2: Develop on-chip power model (OPM) by selecting minimum (Q) signals as input

Part 3: Implement the OPM as part of circuit design

Method: The Basic Power Model



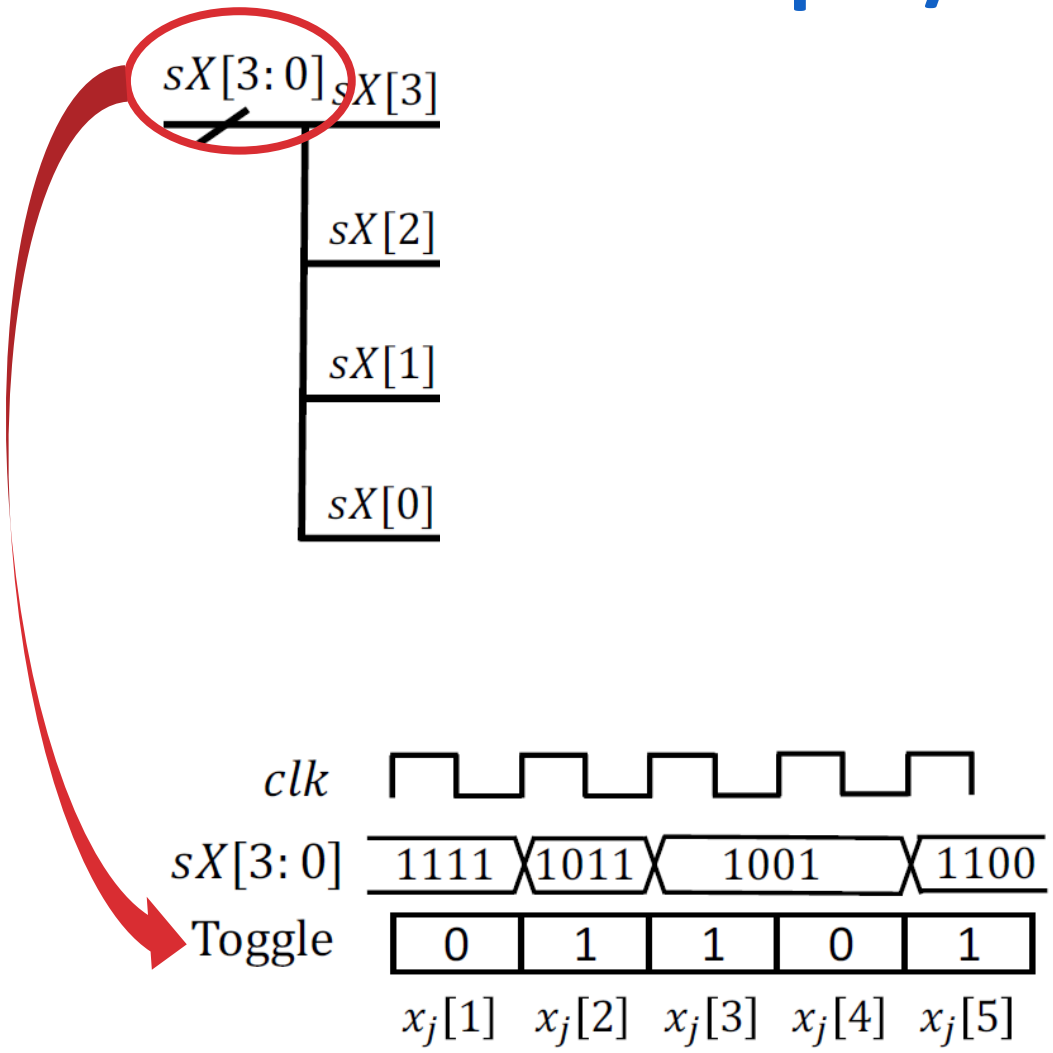
In .fsdb/.vcd file format



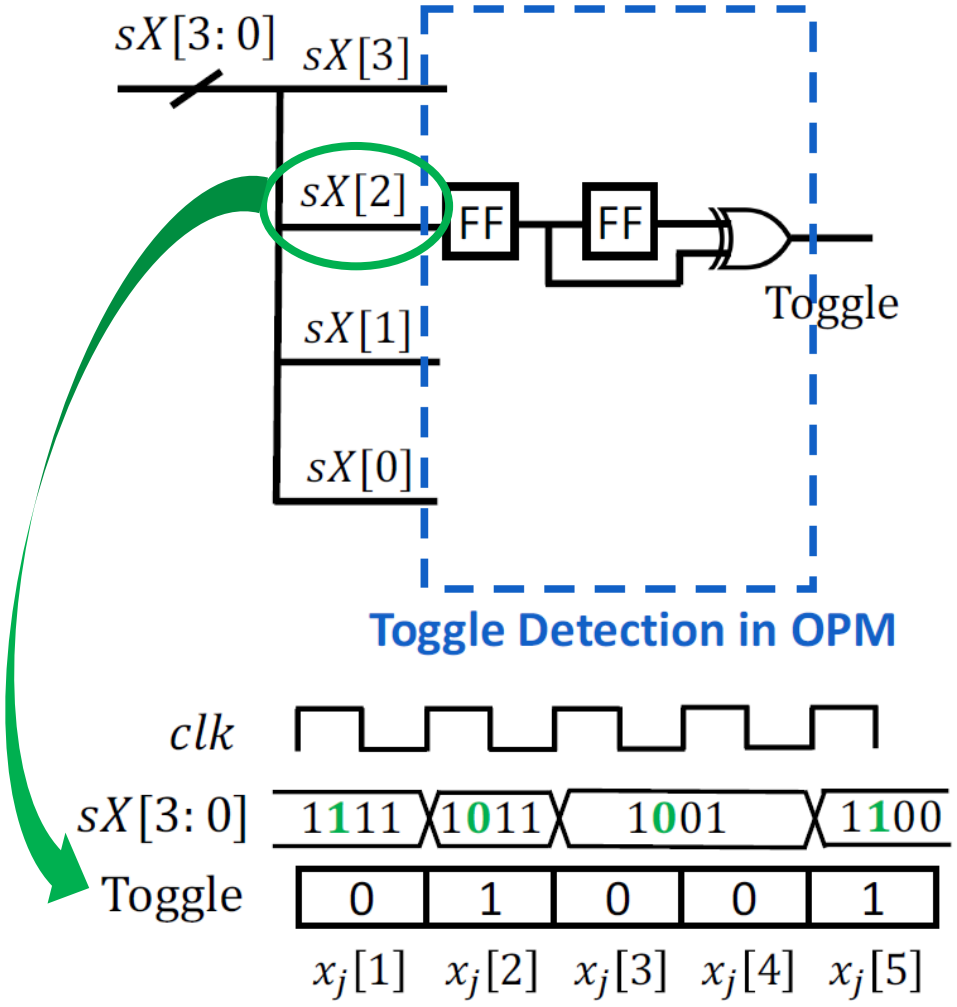
Train the ML model: $F(X) = y$

Input number M **is large.** Only part of them will be selected as input (proxy) for hardware implementation

Method: Model Input/Proxy Candidates



Signal-level selection
Previous method (MICRO'21)



Bit-level selection
This work (DEEP)

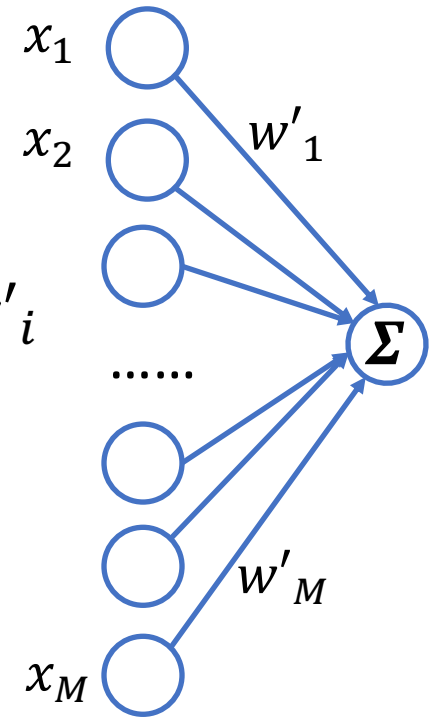
Method: Two-Step Signal Selection Method

Step 1:

- Top-down pruning to **narrow down** the scope of ***M*** variables (V_M) to an intermediate input list with ***I*** variables (V_I)

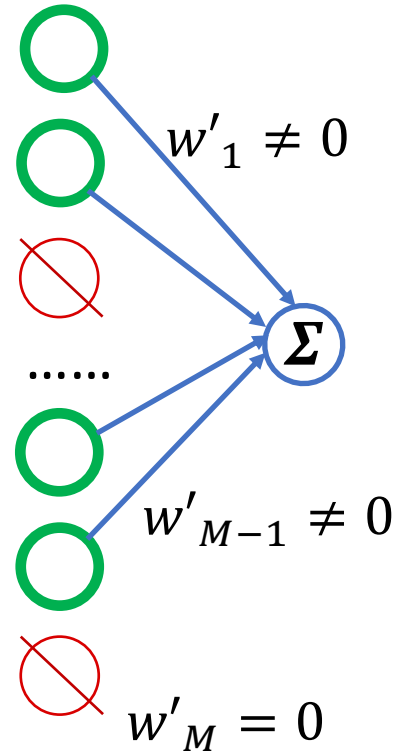
M Features

$$P = \sum_{i=1}^{M} x_i * w'_i$$



Step1: Pruning

Training with **strong** penalty strength



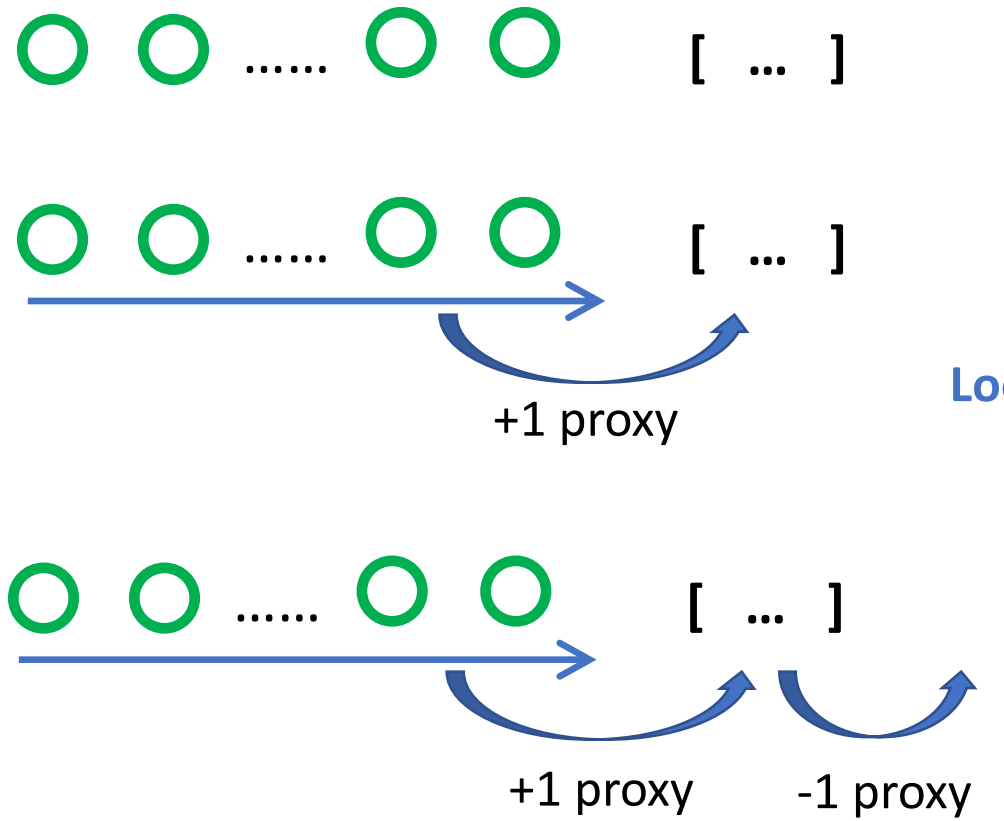
Reach ***I*** non-zero weights and corresponding features

Minimax concave penalty (**MCP**) for pruning

Method: Two-Step Signal Selection Method

Step 2:

- Bottom-up selection of a near-optimal subset of from V_I, as the finalized power model input (proxy) list V_Q



Loop

I feature candidates, an empty selected list

1. Adding:

- Sweep all candidates, find the one adds most accuracy
- Add it to selected list

2. 'Refresh':

- Remove each element from selected list
  - Repeat Step 1 to add one element
- Stop the process until no changes in the list

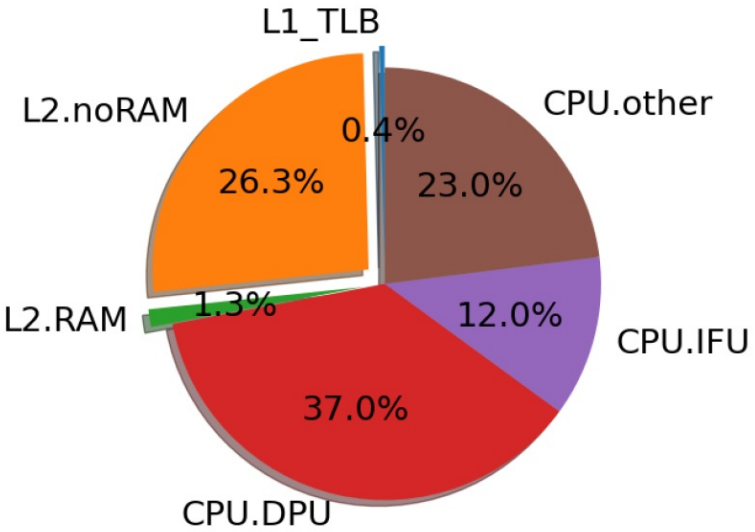
Method: Development of Component-Level Models

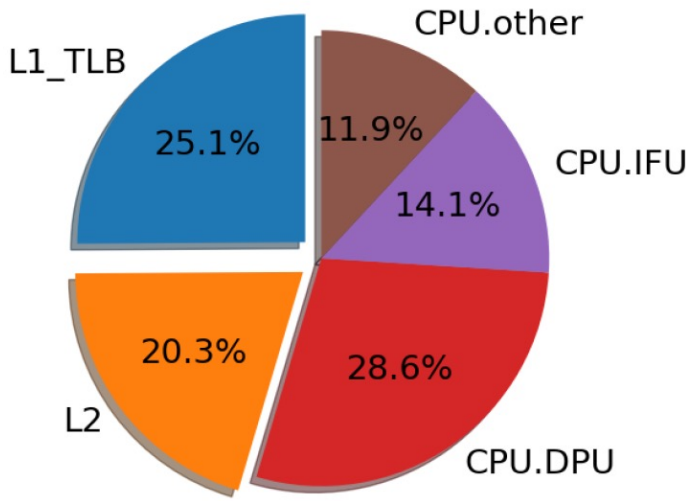
- Sub-OPMs for five selected major submodules in the microprocessor,
  - L1 cache with table lookaside buffer (TLB)
  - L2 cache with the logic maintaining memory coherence,
  - Data processing unit (DPU) of core
  - Instruction fetch unit (IFU) of core
  - All other logic in the CPU core except DPU and IFU
- They also calculate the power of:
  - CPU core
  - CPU core + L1
  - CPU core + L1 + L2 (total power)
- For sub-OPM for the L1 cache, candidates not limited to the L1 cache itself

Result: Experiment Setup and Basic Statistics

#RTL Signal	#Register	#RTL Bit	#Standard Cell	#Macro
155 K	67 K	578 K	603 K	66

Statistics of the micro-processor used in experiment.



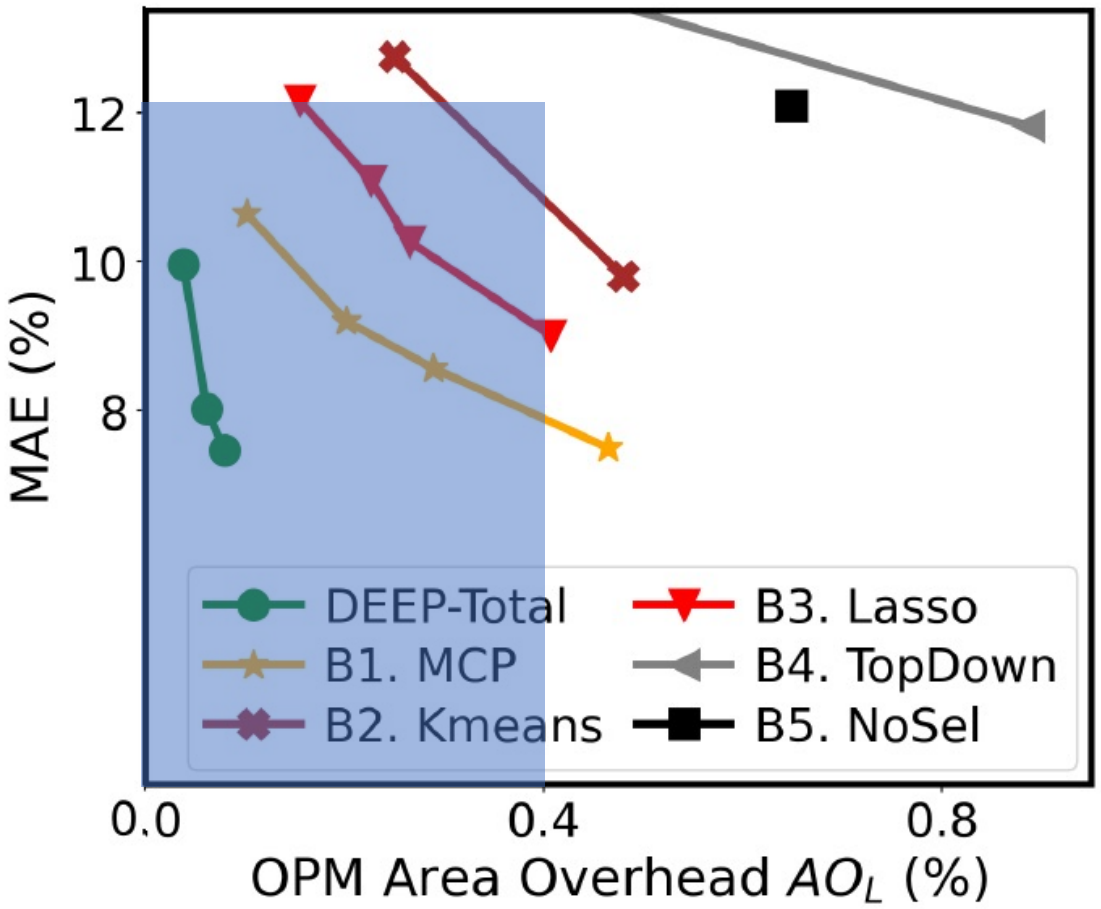


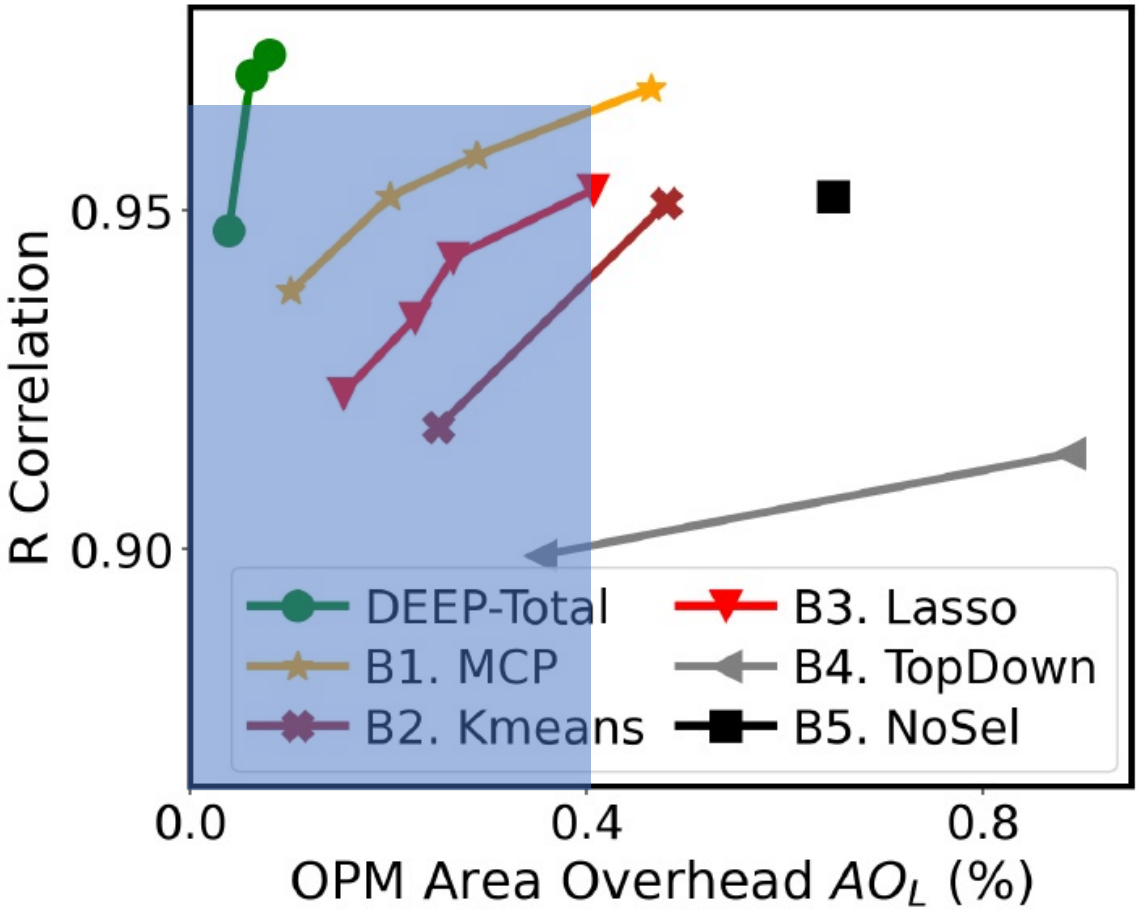
Left: Distribution of all 578 K RTL bits as candidates.

Right: Distribution of power.

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

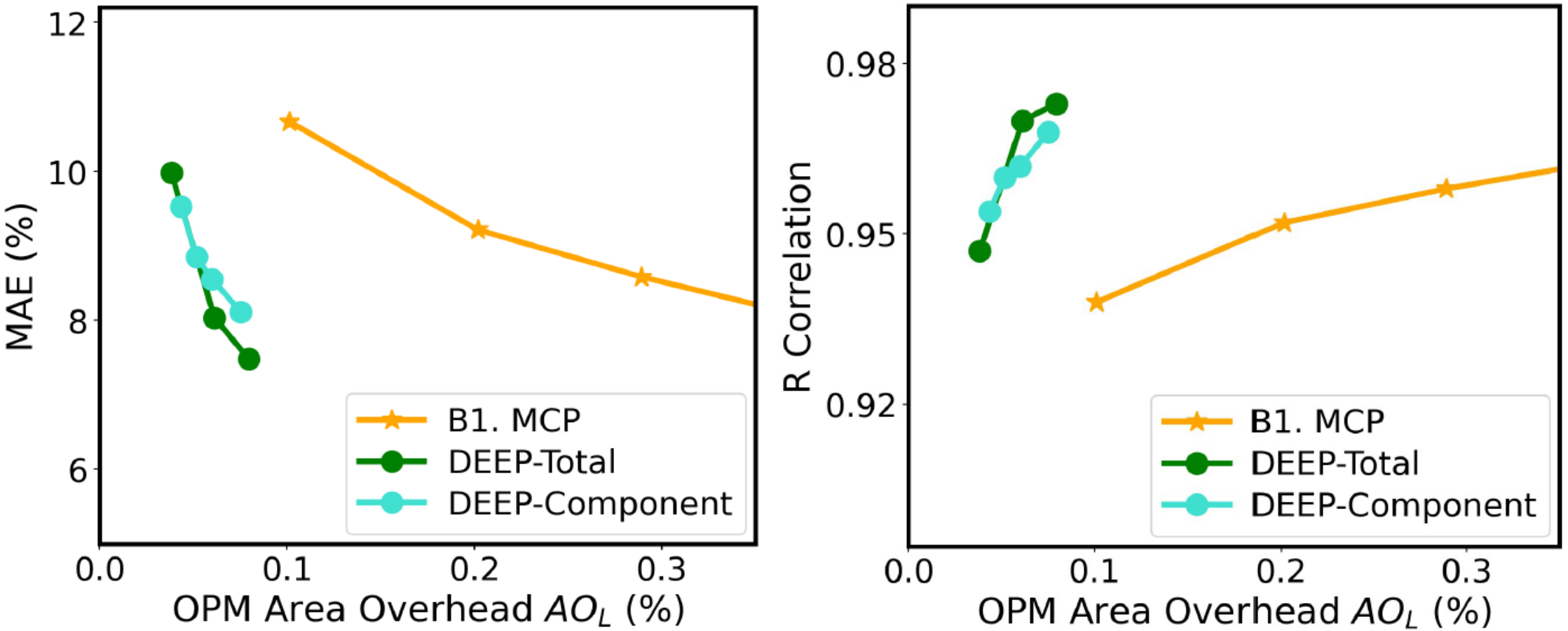
Result: Power Model Development Method Comparison





OPM hardware cost vs. per-cycle power prediction accuracy.

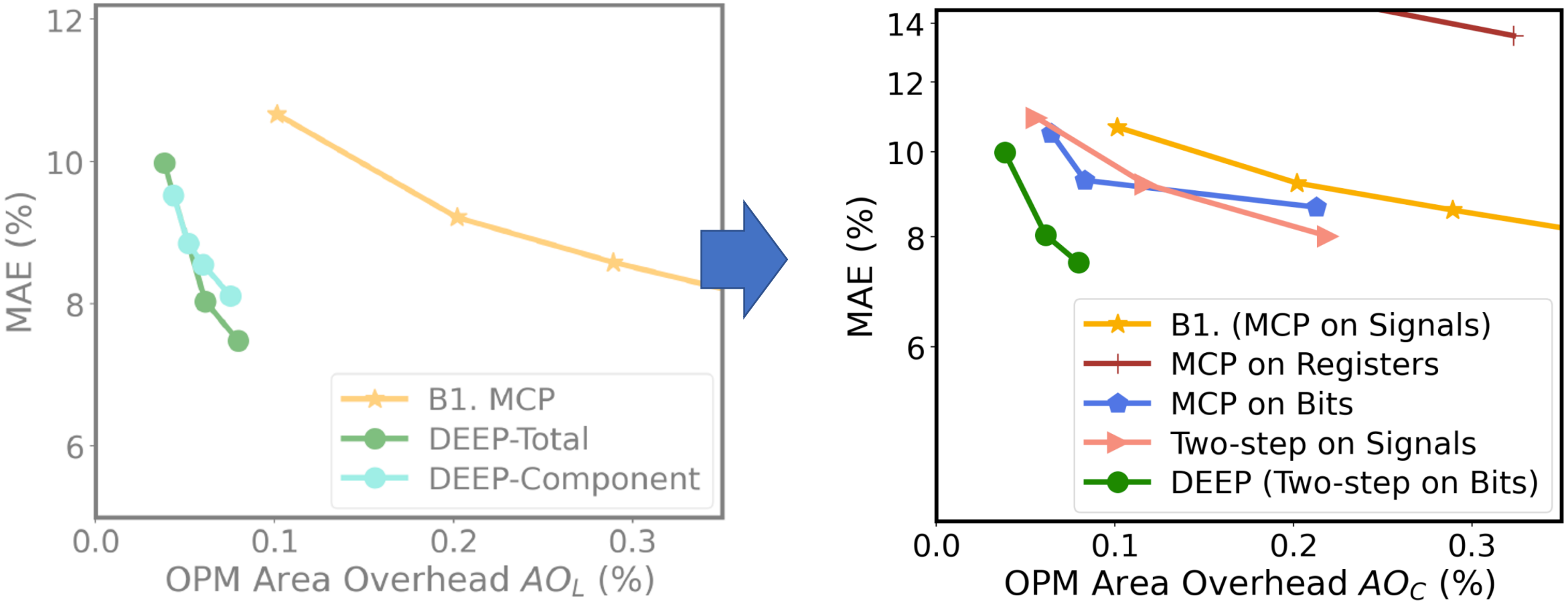
Result: Method Comparison (Zoom in X axis)



OPM hardware cost vs. per-cycle power prediction accuracy.

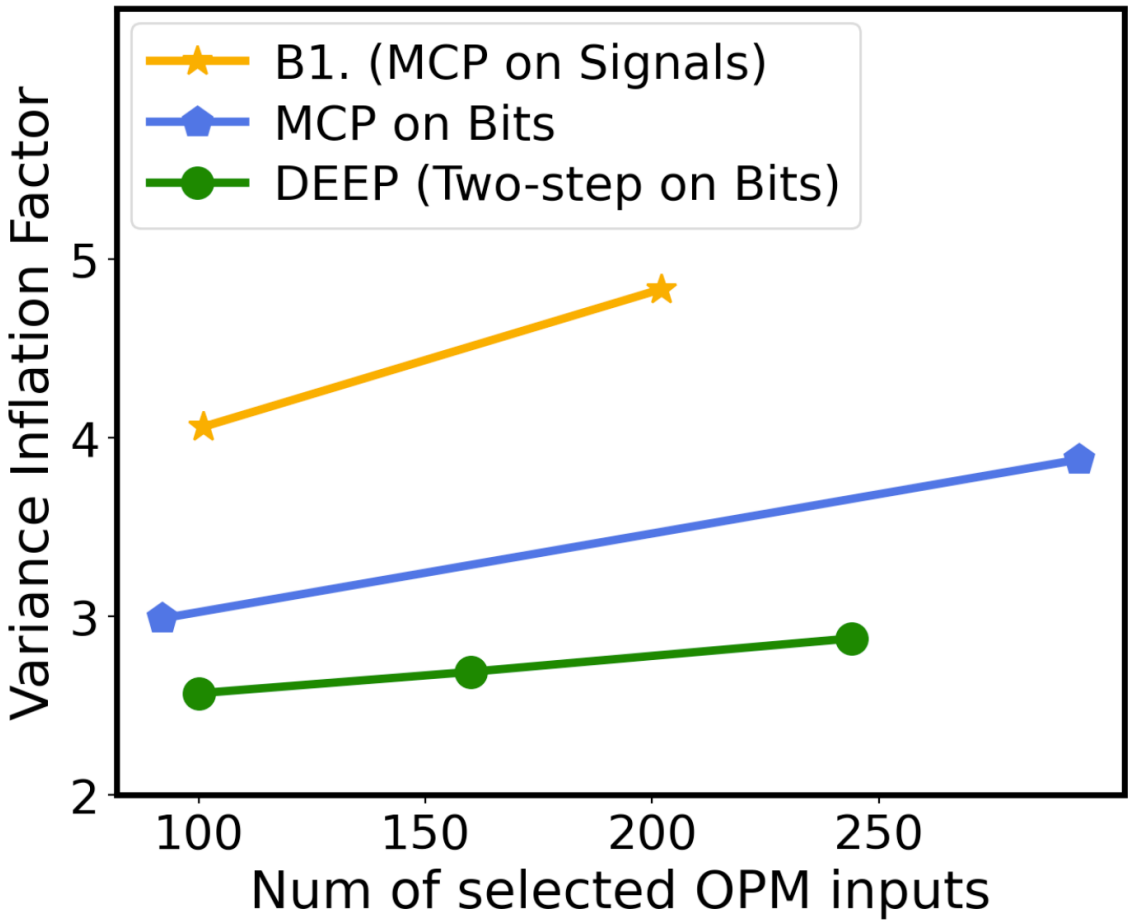
香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Result: Decomposition of DEEP method



OPM hardware cost vs. per-cycle power prediction accuracy.

Analysis of Selected Signals



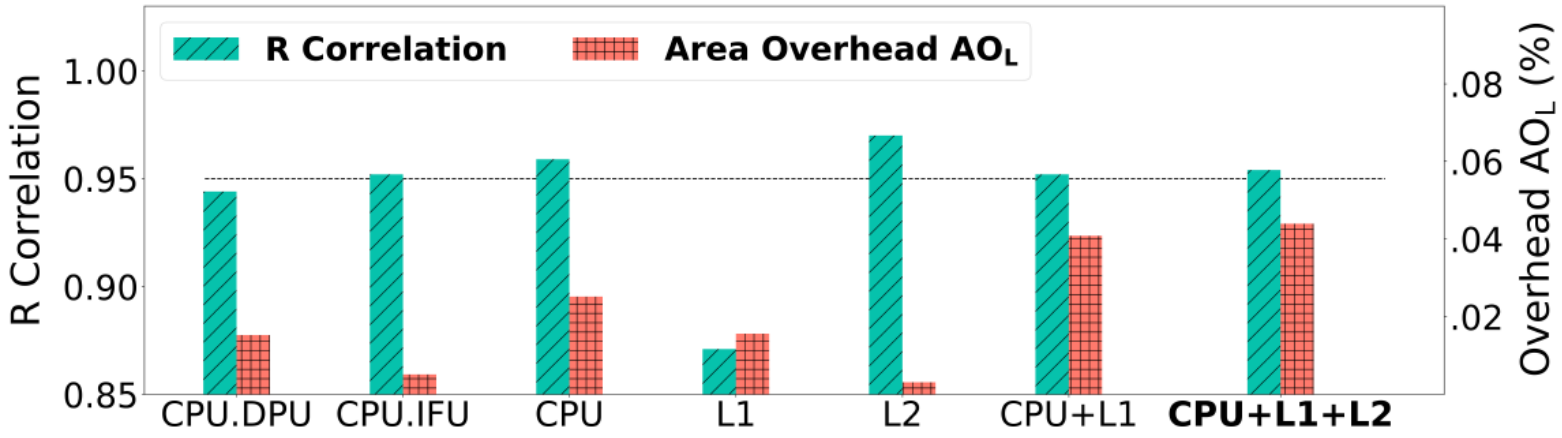
Variance inflation factor (VIF)

Fit each proxy with all other proxies:

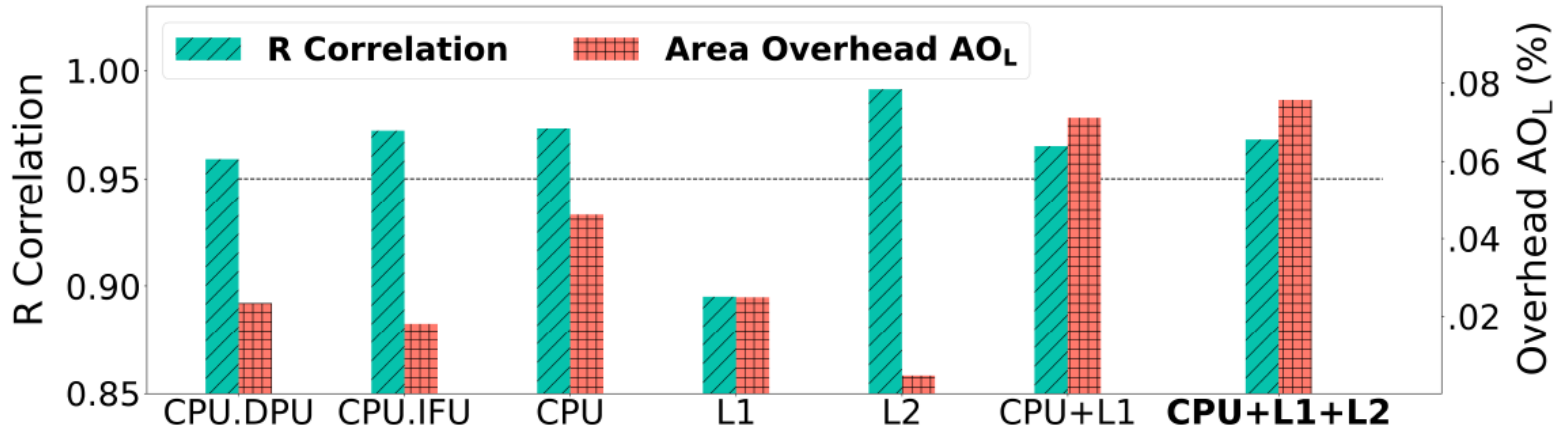
$$\mathrm{VIF} = \frac{1}{1-R^2}$$

Average VIF measures the collinearity among selected proxies.

Result: Component-level OPM Accuracy

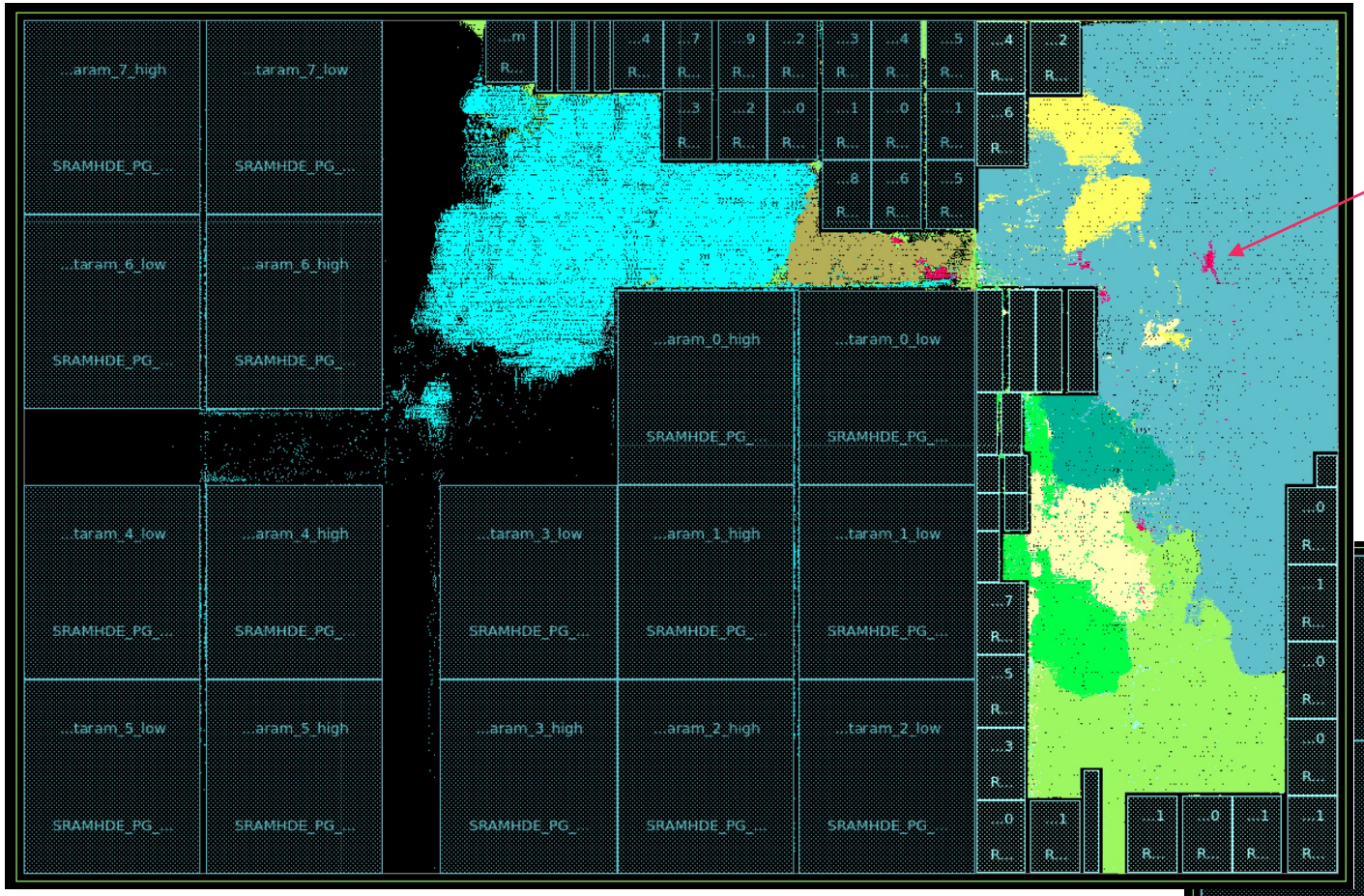


(a) Sub-OPMs with total overhead 0.04%



(b) Sub-OPMs with total overhead 0.08%

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

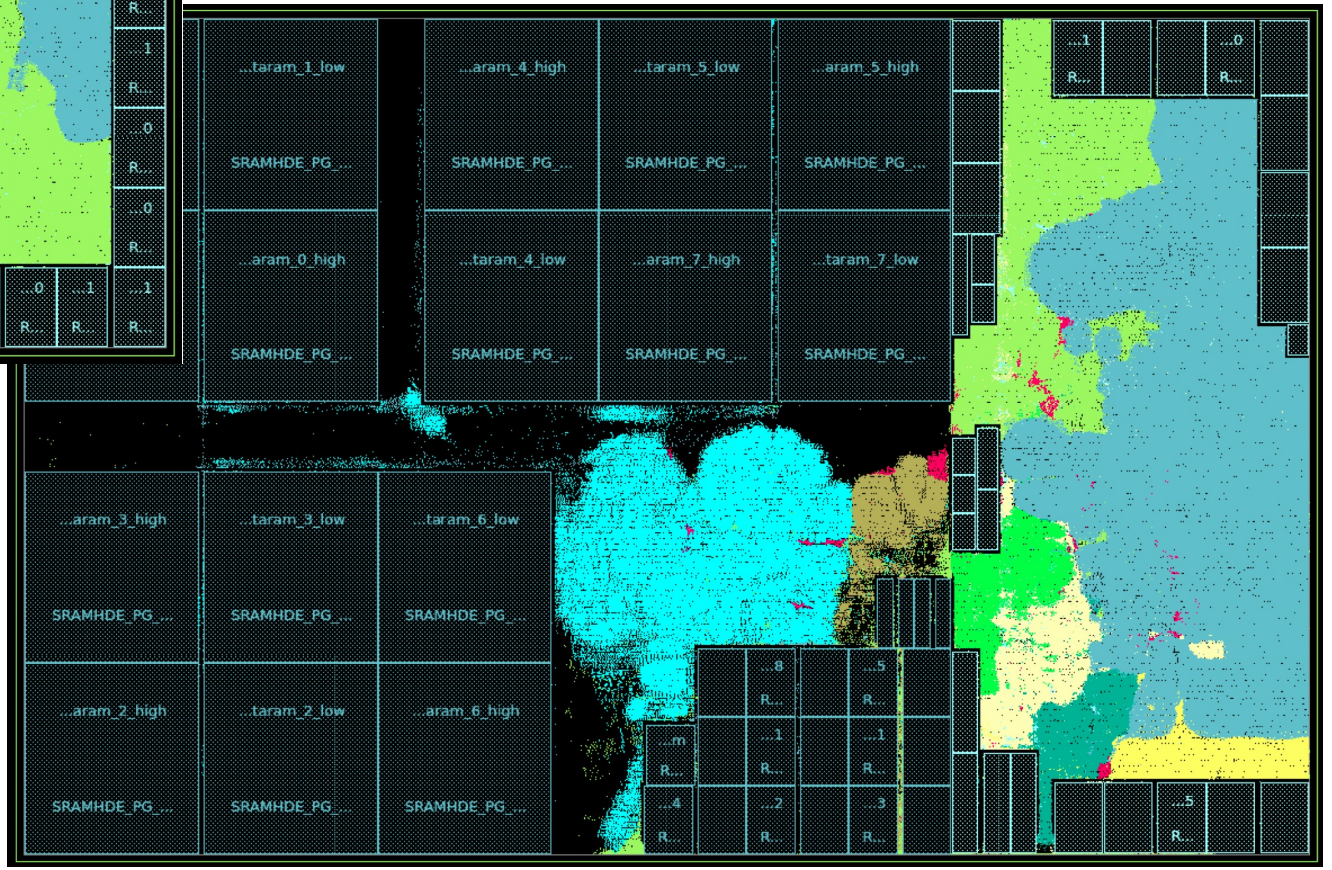


Pink regions are OPM on the layout

Area overhead **0.16%** on this layout.

The MAE = 9.5% and R = 0.951

Baseline from B1 [MICRO'21]

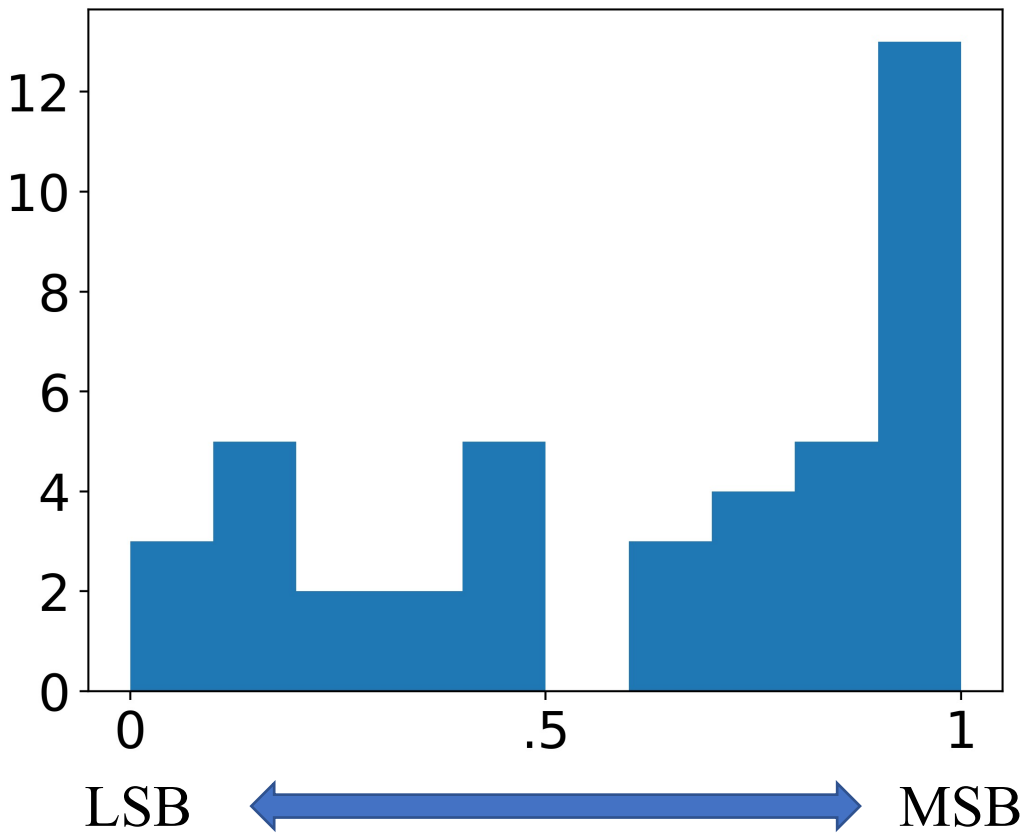


Area overhead **0.04%** on this layout.

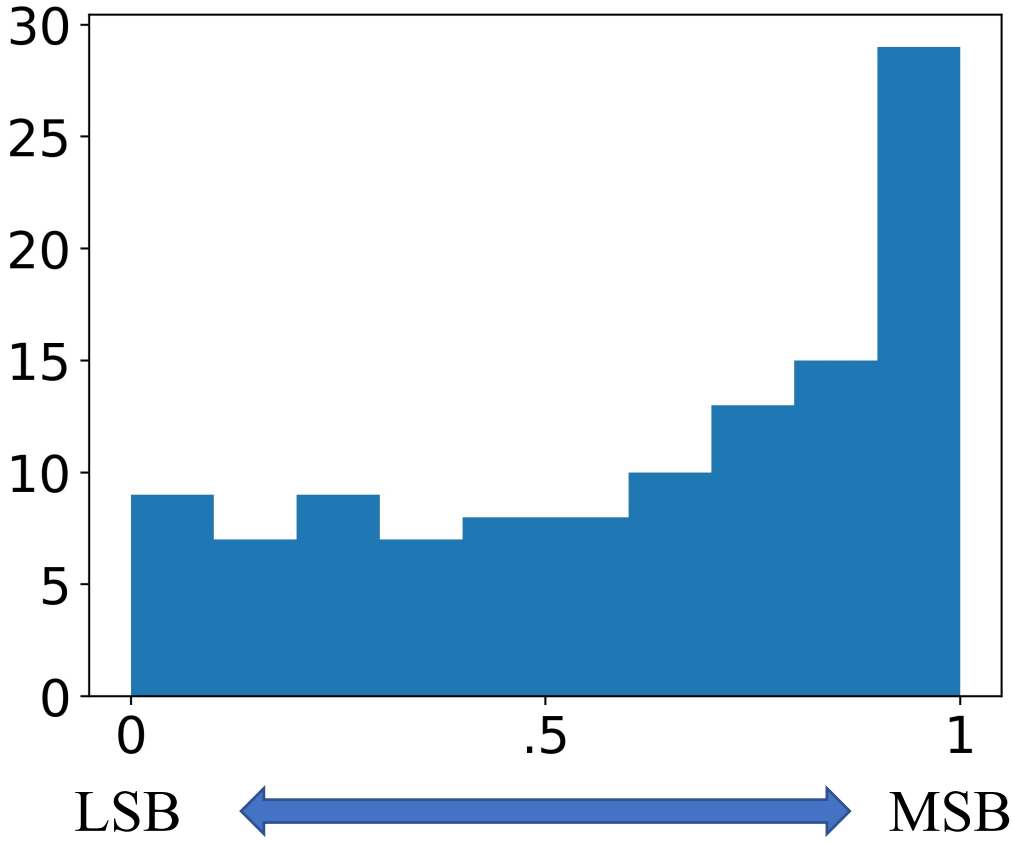
The MAE = 9.5% and R = 0.954.

This DEEP method

Discussion: Histogram of Proxy Bit Position

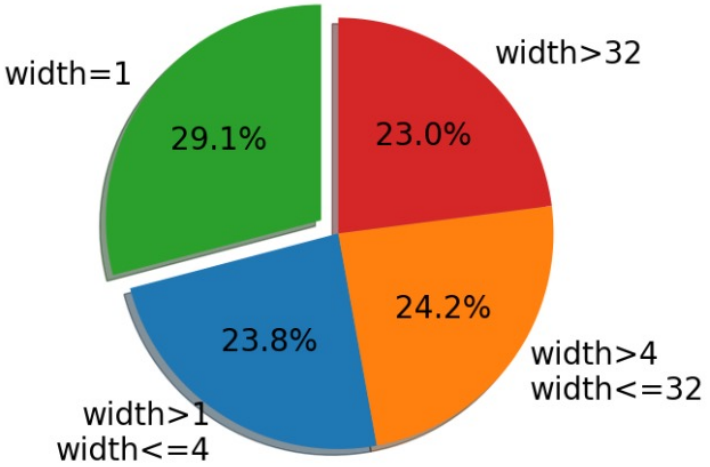


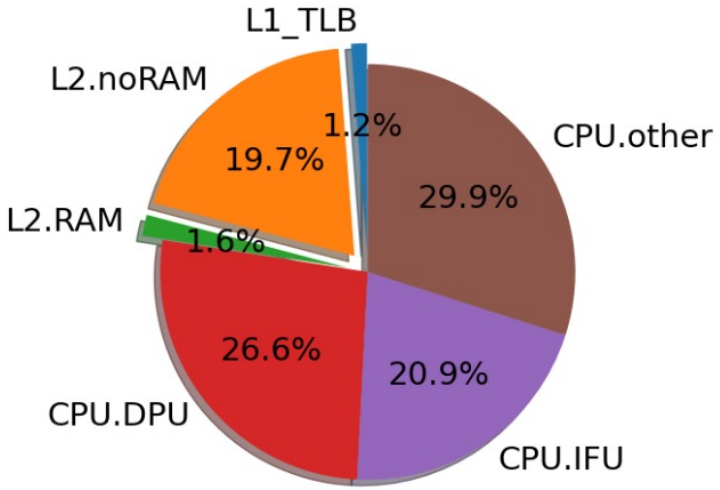
In OPM with overhead 0.04% and R = 0.954.



In OPM with overhead 0.08% and R = 0.968.

Discussion: Analysis of Proxies and Weights





Analysis of 244 selected proxies

- L: Width of their original signals
- R: Where they are selected

DEEP-OPM 1	W bits	1	2	3	4	5	6
	Count	1	2	43	35	15	4
DEEP-OPM 2	W bits	1	2	3	4	5	6
	Count	6	77	120	28	8	5

Post-quantization weight bits distribution

- DEEP-OPM 1 with overhead 0.04%
- DEEP-OPM 2 with overhead 0.08%

Conclusion

- We present a new method for automated OPM development
- It achieves 4 – 6X lower hardware cost over the best baseline, with accuracy $R > 0.97$ with area overhead $< 0.1\%$
- It proposes bit-level and two-stage proxy selection method
- Besides monitoring total power, it reports power of major components without extra cost

Acknowledgement

- Shidhartha Das (Arm Research)
- Xiaoqing Xu (now at Google X)

- NSF-2106828, NSF-2106725
- SRC GRC-CADT 3103.001/3104.001

- Contact us: eezhiyao@ust.hk, chenchia.chang@duke.edu

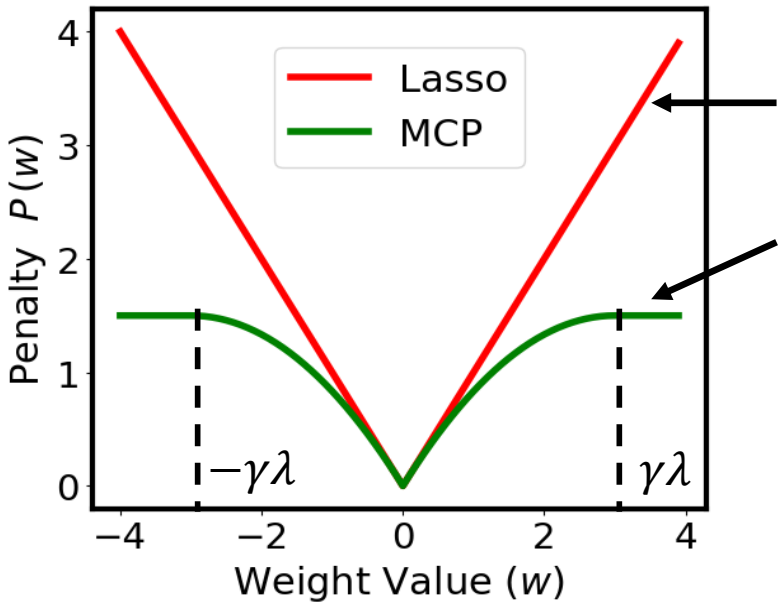
- Thanks! Questions?

香港科技大學
THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY

Backup Slides

Why MCP for Pruning?

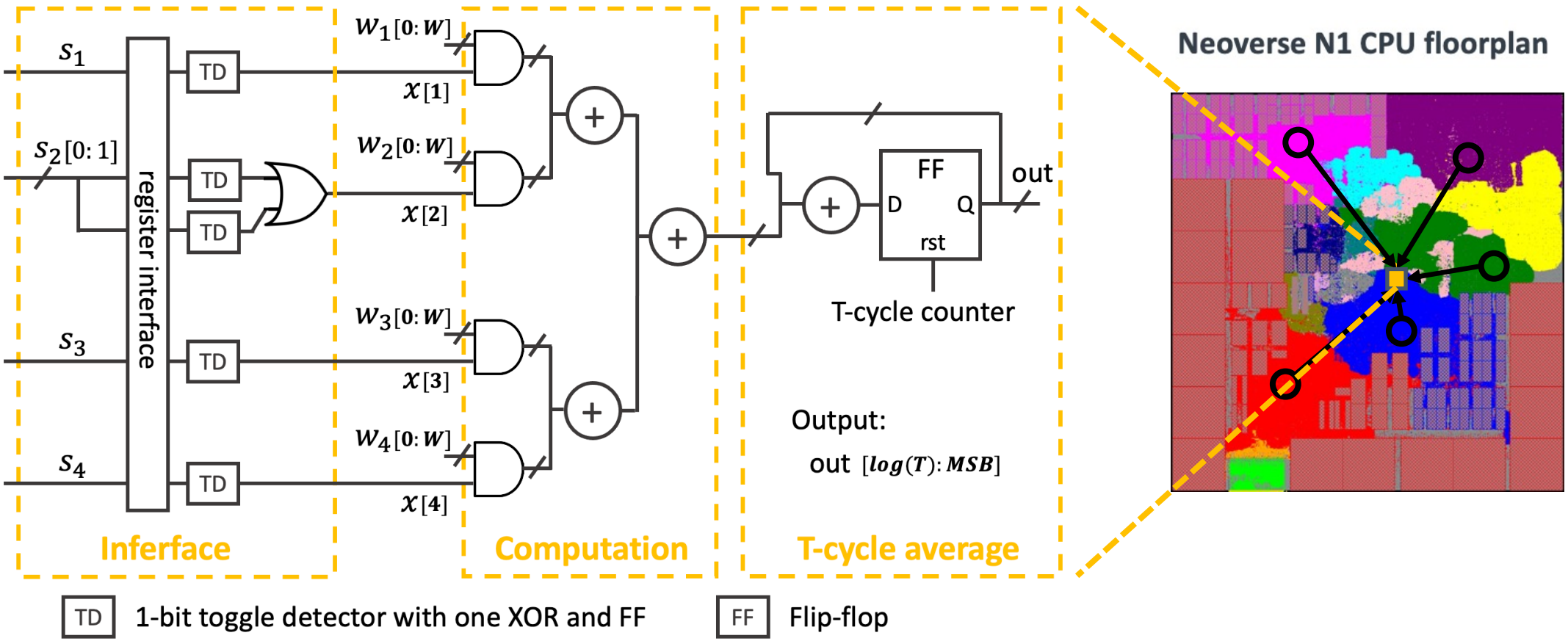
- To make $\boldsymbol{Q} \ll \boldsymbol{M}$, penalty is set to be very large.
- Lasso **degrades** model accuracy under large penalty
- MCP **protects** large weights thus **maintains** model accuracy



Lasso penalize all weights

MCP protects large weights

Overview of the OPM Hardware Design



- **No** multipliers or dividers, only **Q** binary inputs and **W**-bit quantized weights