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

FirePower: Towards a Foundation with Generalizable Knowledge for Architecture-Level Power Modeling

Qijun Zhang, Mengming Li, Yao Lu, and **Zhiyao Xie (Speaker)**

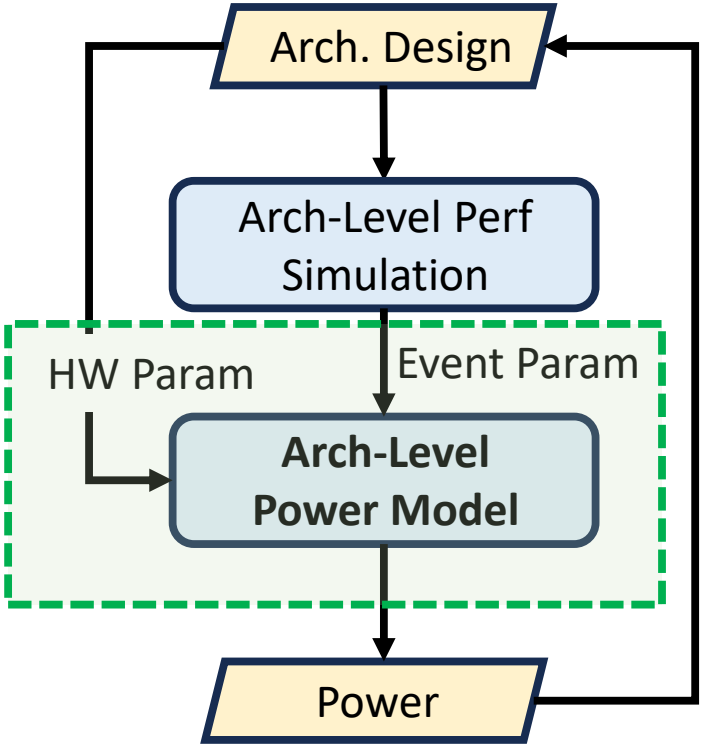
Hong Kong University of Science and Technology

qzhangcs@connect.ust.hk, eezhiyao@ust.hk

Outline

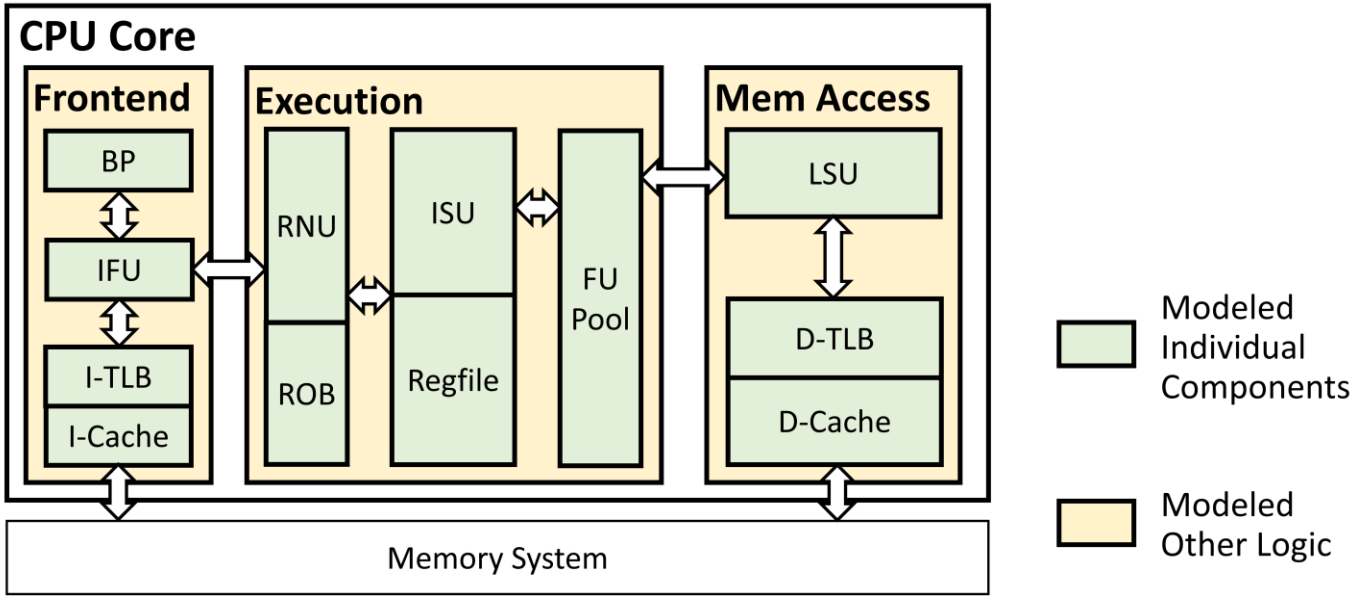
香港科技大學
THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY

Architecture-Level Power Model



- **Power efficiency** is a critical design objective in microprocessor design
- A high demand for **fast**, yet **high-fidelity** architecture-level power modeling
- Input:
  - **Hardware parameters**, e.g. FetchWidth, DecodeWidth, DCacheWays
  - **Event parameters**, e.g. the number of DCache Miss, Branch Misprediction
- Output:
  - **Power**

Components of CPU Core



The architecture of our target Out-of-Order RISC-V CPU core

Configuration and Event Parameters

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

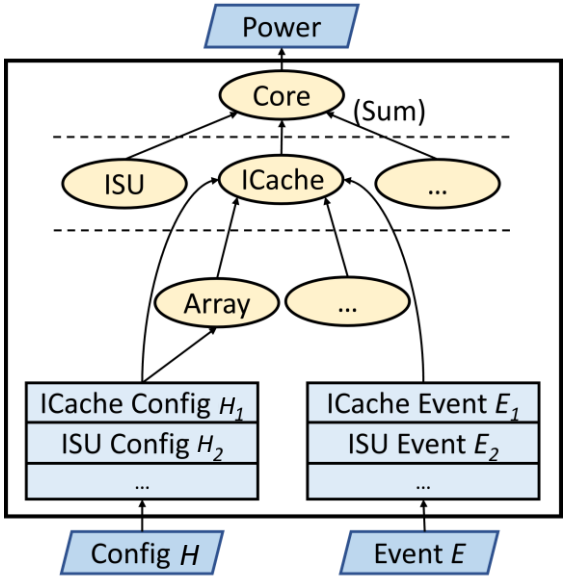
- Major architecture-level **configuration** parameters and **event**:

Component i	Hardware parameters H_i of each component	Event parameters E_i of each component	CPU part
BP	FetchWidth, BranchCount	BTBLookups, condPredicted, condIncorrect, commit.branches	Frontend
IFU	FetchWidth, DecodeWidth FetchBufferEntry, ICacheFetchBytes	fetch.insts, fetch.branches, fetch.cycles, numRefs, numStoreInsts, numInsts, decode.runCycles, decode.blockedCycles, decode.decodedInsts, numBranches, intInstQueueReads, intInstQueueWrites, intInstQueueWakeupAccesses, fpInstQueueReads, fpInstQueueWrites, fpInstQueueWakeupAccesses	
I-TLB	ICacheTLBEntry	itb.accesses, itb.misses	
I-Cache	ICacheWay, ICacheFetchBytes	icache.overallAccesses, icache.overallMisses, icache.ReadReq.mshrHits, icache.ReadReq.mshrMisses, icache.tagAccesses	
RNU	DecodeWidth	intLookups, renamedOperands, fpLookups, renamedInsts, runCycles, blockCycles, committedMaps	Execution
ROB	DecodeWidth, RobEntry	rob.reads, rob.writes	
ISU	DecodeWidth, MemIssueWidth, FpIssueWidth, IntIssueWidth	IssuedMemRead, IssuedMemWrite, IssuedFloatMemRead, IssuedFloatMemWrite, IssuedIntAlu, IssuedIntMult, IssuedIntDiv, IssuedFloatMult, IssuedFloatDiv	
Regfile	DecodeWidth, IntPhyRegister, FpPhyRegister	intRegfileReads, fpRegfileReads, intRegfileWrites, fpRegfileWrites, functionCalls	
FU Pool	MemIssueWidth, FpIssueWidth, IntIssueWidth	intAluAccesses, fpAluAccesses	
LSU	LDQEntry, STQEntry, MemIssueWidth	MemRead, InstPrefetch, MemWrite	Mem Access
D-TLB	DCacheTLBEntry	dtb.accesses, dtb.misses	
D-Cache	DCacheWay, DCacheTLBEntry, DCacheMSHR, MemIssueWidth	dcache.ReadReq.accesses, dcache.WriteReq.accesses, dcache.ReadReq.misses, dcache.WriteReq.misses, dcache.overallMisses, dcache.MshrHits, dcache.MshrMisses, dcache.tagAccesses	
Other Logic	All	All	Other Logic

Assume there are N components, for **component i**:

1) **Hardware** parameters denoted as H_i, with $H = \{H_i \mid i \in [1, N]\}$
2) **Event** parameters denoted as E_i, with $E = \{E_i \mid i \in [1, N]\}$

Analytical Model



- Example: McPAT [MICRO'09]
- Unreliable accuracy

Analytical Model

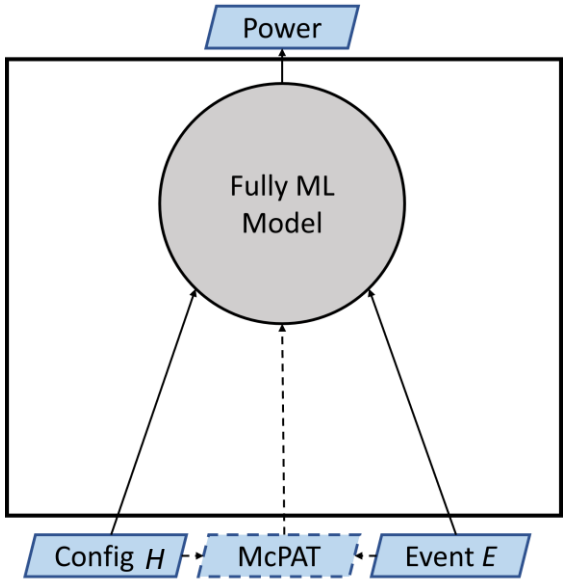
- **Explicitly** design separate **analytical** models for each **component**
- For example, the model of ICache is :

$$P^i_{\text{ana}} = \frac{\#Hit * Energy\ per\ hit + \#Miss * Energy\ per\ miss}{Total\ benchmark\ execution\ time}$$

- where

$$Energy\ per\ hit/miss = \boldsymbol{F}^i(H_i)$$

ML-based Model



- Example: McPAT-Calib [ICCAD'21]
- Rely on sufficient data

ML-based Model

- The F_{ml} denotes data-driven ML methods
- P_{ml} denotes the power prediction value
- Can be formulated below:

$$P_{\mathrm{ml}} = \boldsymbol{F_{ml}}\left(\{H_i \mid i \in [1, N]\},\ \{E_i \mid i \in [1, N]\}\right)$$

Prior Works

- **ML-based Model:**
  - PowerTrain (ISLPED'15)
  - McPAT-Calib (ICCAD'21)
  - Transfer Learning (ASP-DAC'23)

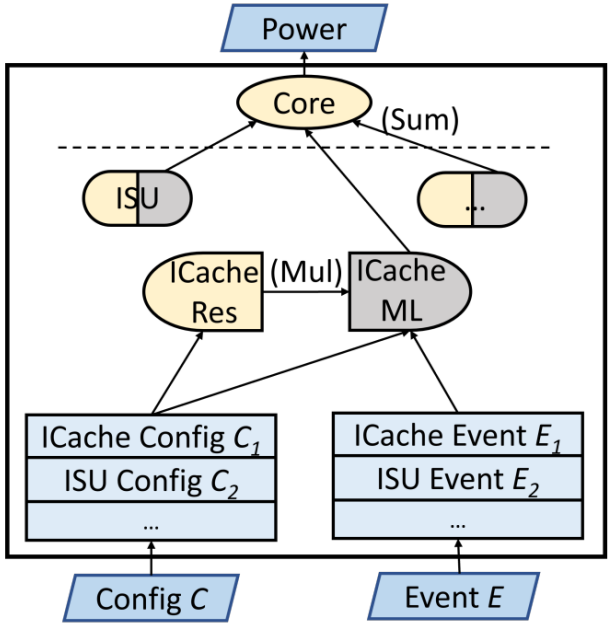
- **Problem of ML-based Model:**

  (1) Unapplicable to a **different architecture**

  (2) Bad accuracy with **limited training data**

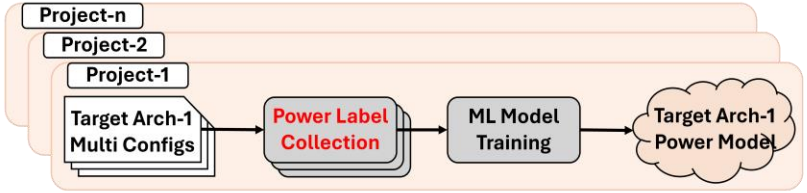
PANDA

- Our another work, PANDA, unifies these two methods
- Require **human-effort**, out of the scope of this work

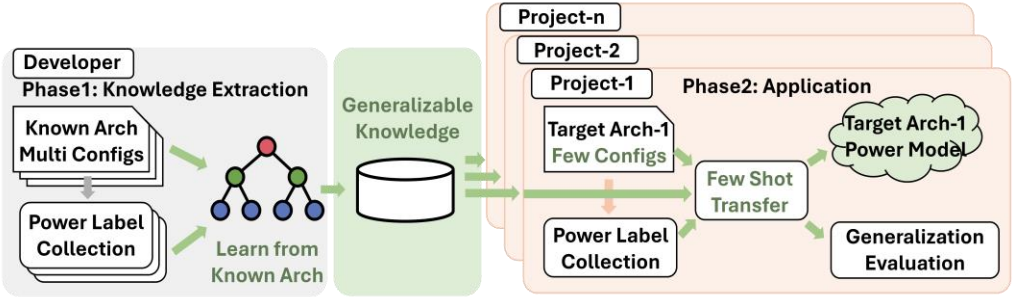


FirePower

- ***Goal:*** Target the **few-shot** learning scenario for **new** target architectures
  - E.g. Train on ***BOOM*** *(known arch),* apply to ***XiangShan*** *(new target arch)*
- ***Key Idea:*** Extract general knowledge from known architecture as a "foundation" to support modeling new architectures



(a) Mainstream Architecture-Specific Power Modeling Paradigm



(b) New Paradigm based on a Foundation with Extracted Generalizable Knowledge

Outline

FirePower

- **Insight 1**:
- Model power for each **power-friendly design component**.

- **Insight 2:**
- Some knowledge is more **general**, while others are more **architecture-specific**.

Power Model Formulation

- **Insight 1**:
- -> Choose component-level power modeling
- **Insight 2:**
- -> Decouple each component's power into generalizable and architecture-specific parts

- The FirePower power model can be formulated below:

$$P^i = \boldsymbol{F_{hw}}^i(H_i) * \boldsymbol{F_{event}}^i(H_i, E_i)$$

Power Model Formulation

- The FirePower power model can be formulated below:

$$P^i = {F_{hw}}^i(H_i) * {F_{event}}^i(H_i, E_i)$$

- ${F_{hw}}^i$ denotes the **hardware model** of component i, which learns the basic correlation between hardware scale and hardware parameters.

- ${F_{event}}^i$ denotes the **event model**, an ML model to capture more complex correlations related to event statistics

Power-Friendly Component Definition

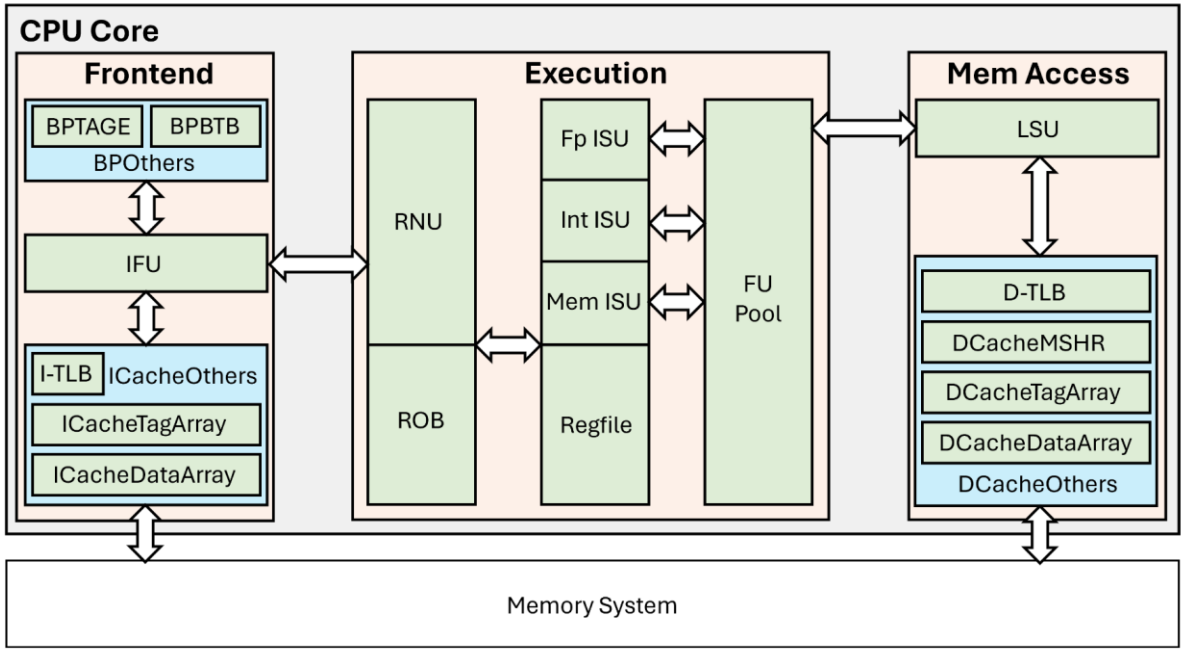
香港科技大學
THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY

Two **targets** in defining power-friendly component:

- **1) Common**:
- Each component can be found in different out-of-order CPU architectures

- **2) Fine-grained**:
- Circuit in the same component should correlate with similar hardware parameters and event statistics

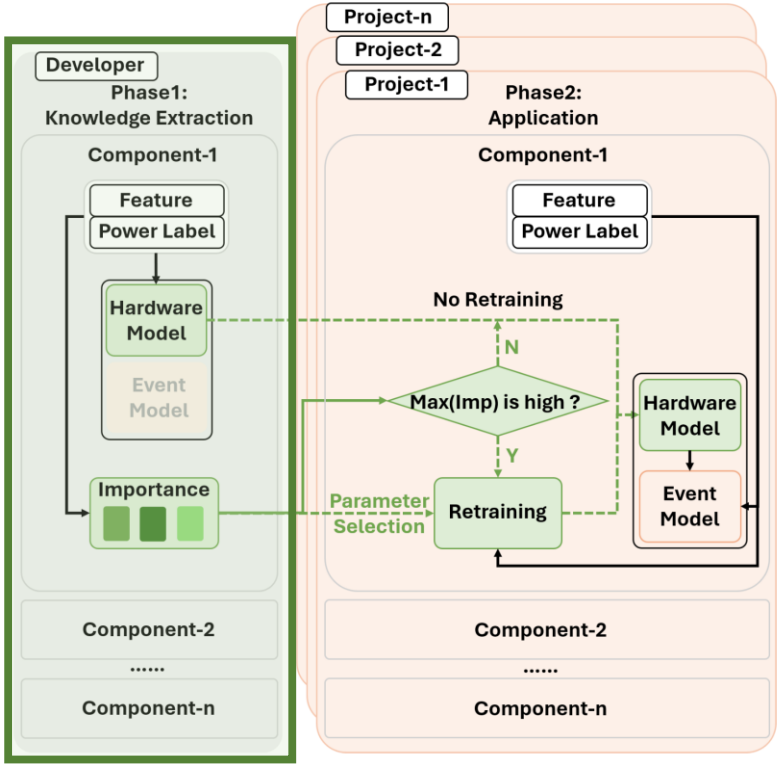
Power-Friendly Component Definition



the proposed component definition of our target Out-of-Order CPU core

Outline

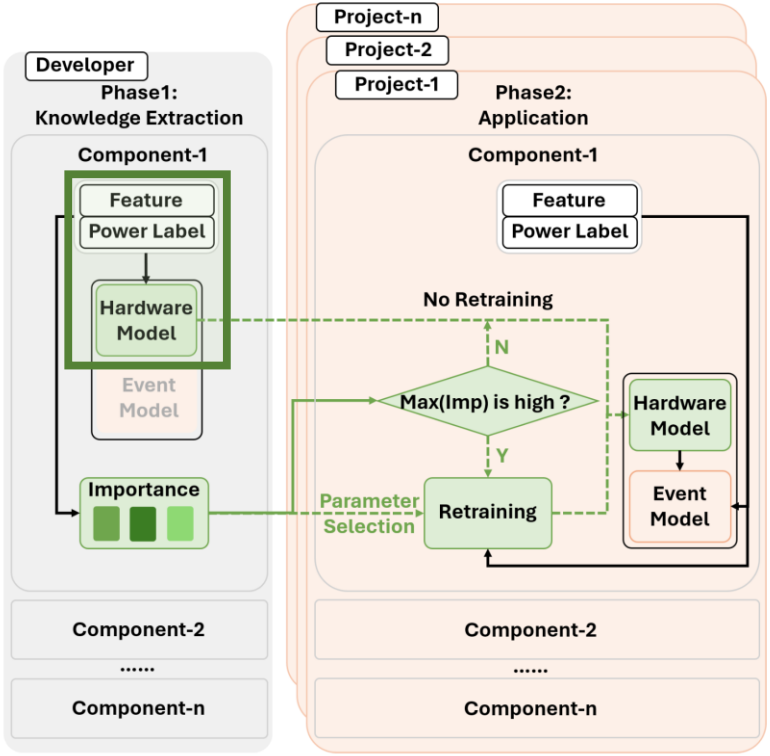
FirePower Framework: Phase 1



Phase 1: Knowledge Extraction

- Knowledge extraction in phase 1 extracts:
  - hardware model
  - parameter importance

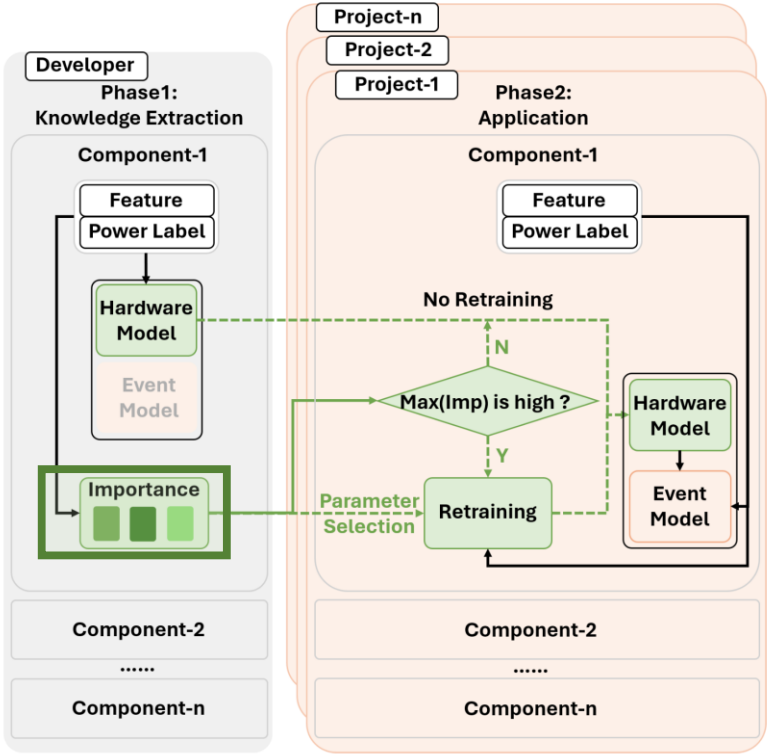
Phase 1 Knowledge Extraction: Hardware Model



- Hardware Model is an ML model

- Labels:
  - The average power across all workloads
  - reflect the general power characteristics across workloads
- Input features:
  - hardware parameters H_i of each component
- ML model:
  - XGBoost

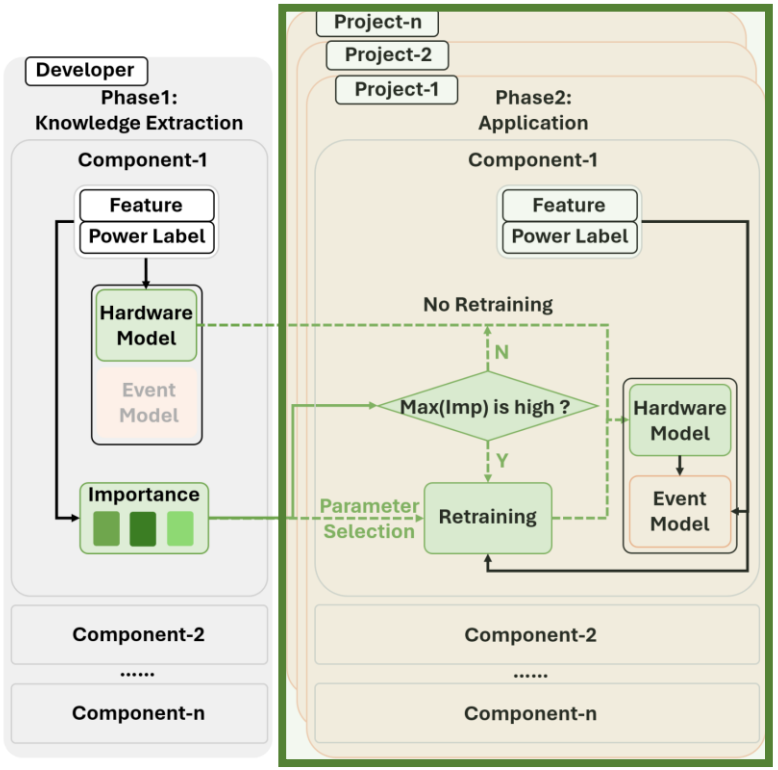
Phase 1 Knowledge Extraction: Parameter Importance



- Parameter importance is the importance of hardware parameters for each component
- Key idea:
  - Evaluate whether there is a dominating hardware parameter for each component

Outline

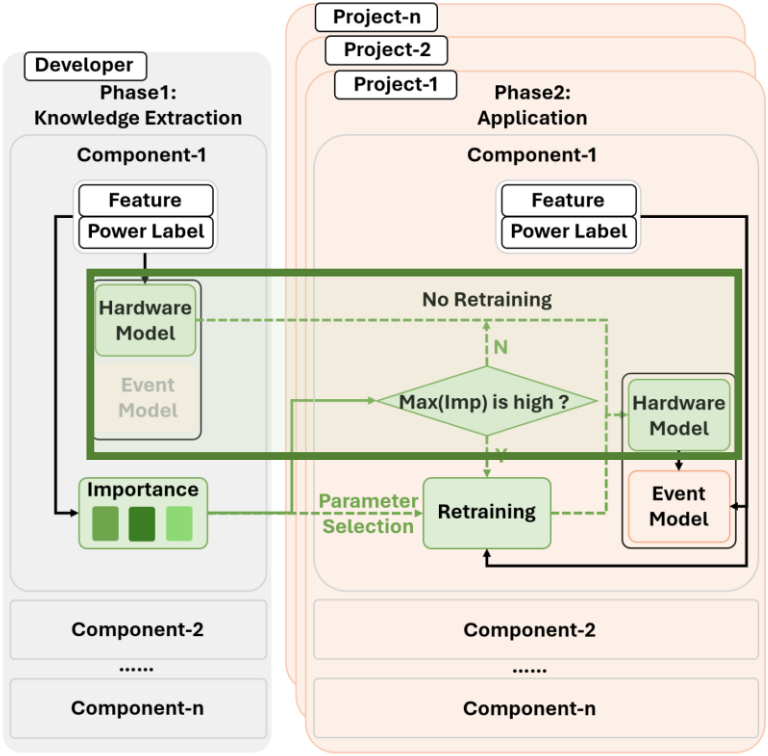
FirePower Framework: Phase 2



Phase 2: Apply to new architecture

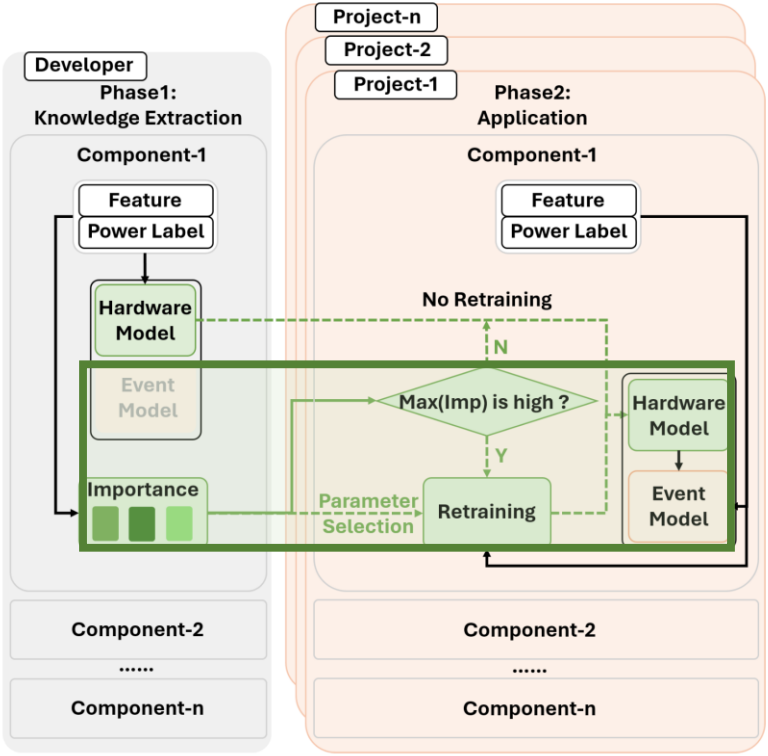
- Adopts two knowledge generalization strategies:
  - Retraining
  - No Retraining

Phase 2 Apply to New Architecture: No Retraining



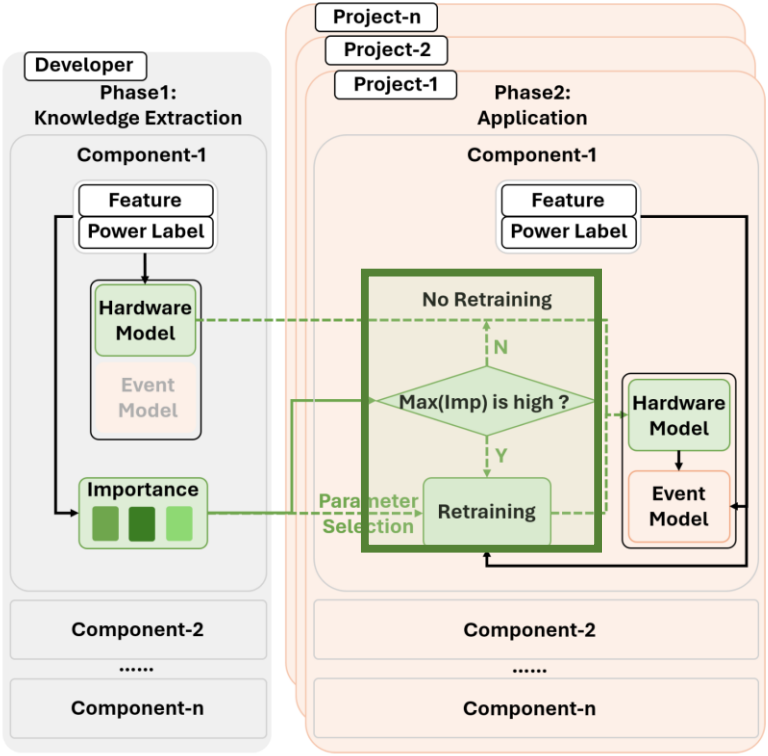
- Method:
  - Adopts the hardware model trained on the known architecture dataset
- Discussion:
  - May not accurately estimate average power
- Rationale:
  - Captures correlation rather than absolute value
  - Ratio will be captured by event model

Phase 2 Apply to New Architecture: Retraining



- Method:
- Retrain a new hardware model on new target architecture:
  - Feature: Only the important feature
  - ML model: Linear model
- Rationale:
  - Simplify the model to avoid overfitting
  - Suitable for components with simple power characteristics

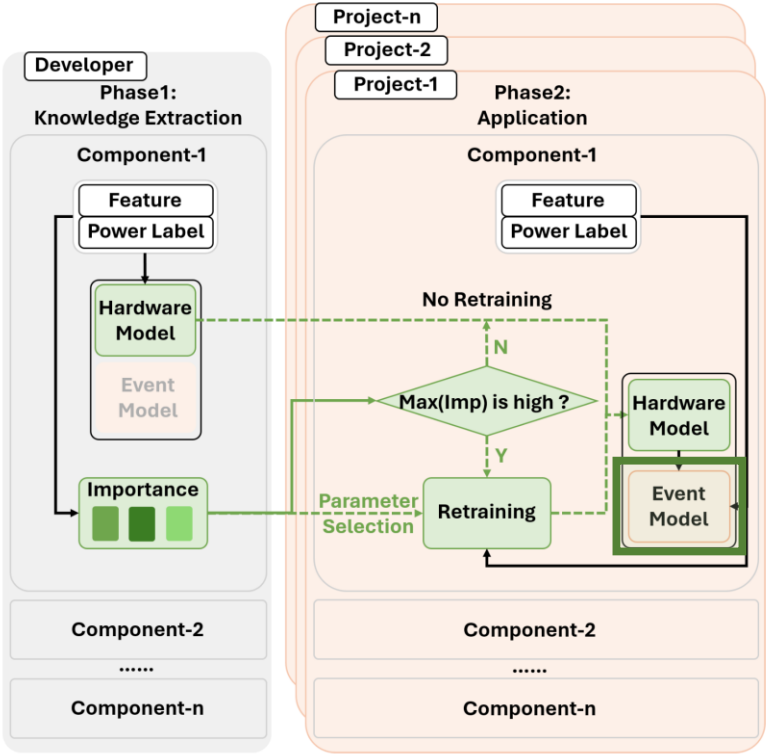
Phase 2 Apply to New Architecture: Strategy Selection



- If max importance > threshold:
  - Retraining
- Otherwise:
  - No Retraining

- Rationale:
  - Dominating importance indicates simple overall power correlation
  - Uniform distribution means complexity

Phase 2 Apply to New Architecture: Event Model



Event Model is an ML model

- Labels:
  - the ratio between the component power label and the hardware model
- Input features:
  - hardware parameters H_i and event statistics E_i of each component
- ML model:
  - XGBoost

Outline

Generalization Quality Evaluation

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

- FirePower supports evaluating the generalization quality
  - Help to determine whether to accept the generalized model or not

- Method:
  - Compare the average power label with the prediction of the hardware model obtained from phase 1 on accessible configs
  - High accuracy indicates a high generalization quality

Outline

Experiment – Setup

- **15** RISC-V BOOM configurations, **10** XiangShan (XS) configurations
  - Configuration: a CPU design with a specific set of hardware parameters
- 8 commonly used testbenches
- Two scenarios: BOOM->XS, XS->BOOM (train on BOOM, apply to XS, and vice versa)

Hardware Parameter	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10
FetchWidth	4	4	4	4	4	8	8	8	8	8	8	8	8	8	8	4	4	4	4	4	8	8	8	8	8
DecodeWidth	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5	2	2	2	3	3	3	4	4	4	5
FetchBufferEntry	5	8	16	8	16	24	18	24	30	24	32	40	30	35	40	8	16	24	16	24	24	24	32	32	24
RobEntry	16	32	48	64	64	80	81	96	114	112	128	136	125	130	140	16	32	48	64	64	80	81	96	114	112
IntPhyRegister	36	53	68	64	80	88	88	110	112	108	128	136	108	128	140	36	53	68	64	80	88	88	110	112	108
FpPhyRegister	36	48	56	56	64	72	88	96	112	108	128	136	108	128	140	36	53	68	64	80	88	88	110	112	108
LDQ/STQEntry	4	8	16	12	16	20	16	24	32	24	32	36	24	32	36	16	20	24	20	24	28	24	32	40	32
BranchCount	6	8	10	10	12	14	14	16	16	18	20	20	18	20	20	7	7	7	7	7	7	7	7	7	7
Mem/FpIssueWidth	1	1	1	1	1	1	1	1	2	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
IntIssueWidth	1	1	1	1	2	2	2	3	3	4	4	4	5	5	5	2	2	2	2	4	4	4	6	6	6
DCache/ICacheWay	2	4	8	4	4	8	8	8	8	8	8	8	8	8	8	4	4	8	4	4	8	8	8	8	8
DTLBEntry	8	8	16	8	8	16	16	16	32	32	32	32	32	32	32	8	8	16	8	8	16	16	16	32	32
MSHREntry	2	2	4	2	2	4	4	4	4	4	4	8	8	8	8	2	2	4	2	2	4	4	4	4	4
ICacheFetchBytes	2	2	2	2	2	4	4	4	4	4	4	4	4	4	4	2	2	2	2	2	2	2	2	2	2

Experiment – Baseline and Ablation Study

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

- Baseline:

  (a) McPAT-Calib

- Four Ablation Studies:

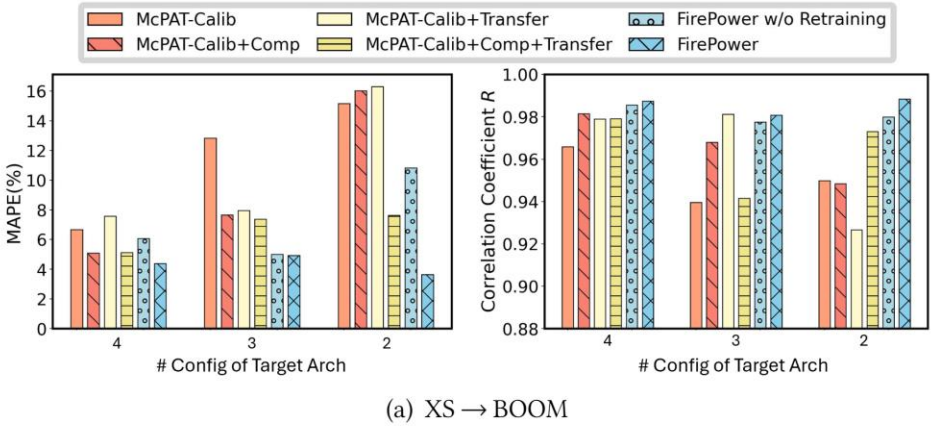
  (b) The McPAT-Calib + Component: ML models for **each component**

  (c) The McPAT-Calib + Transfer Learning: A source model on the known architecture, and then adopt **transfer learning algorithms, pseudo label**

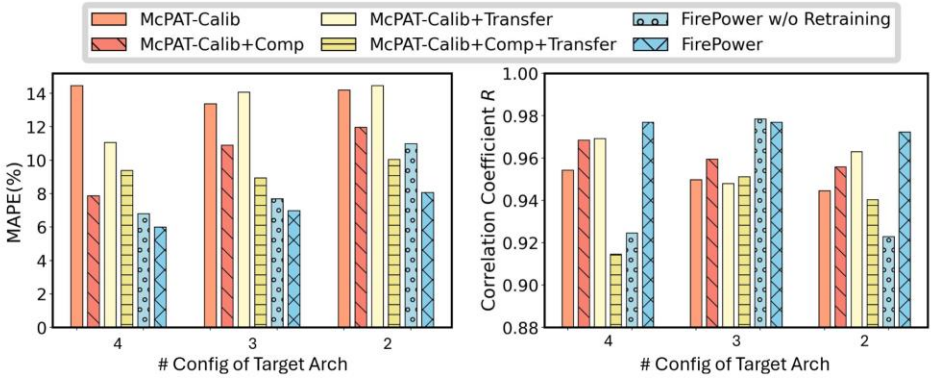
  (d) The McPAT-Calib + Component + Transfer Learning: **Combine (b) and (c)**

  (e) FirePower without Retraining: **Only** adopt the **No Retraining**

Power Modeling Results



(a) XS → BOOM



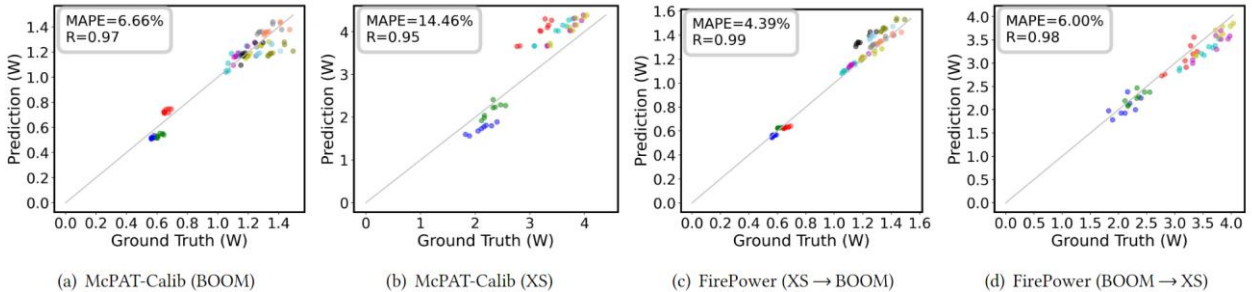
(b) BOOM → XS

- #Config of ***new target*** arch for training: 4, 3, and 2
  - Few-shot scenarios
- Report MAPE and correlation R
- **Outperform** other methods in **all** scenarios

Power Modeling Results

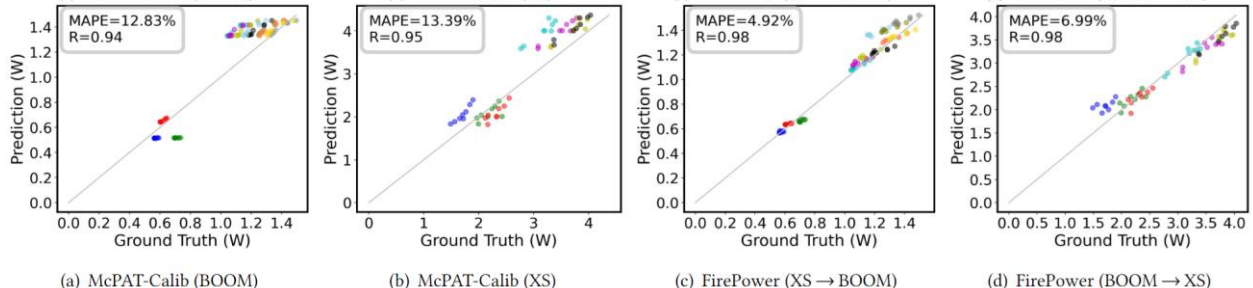
香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Known 4 Config



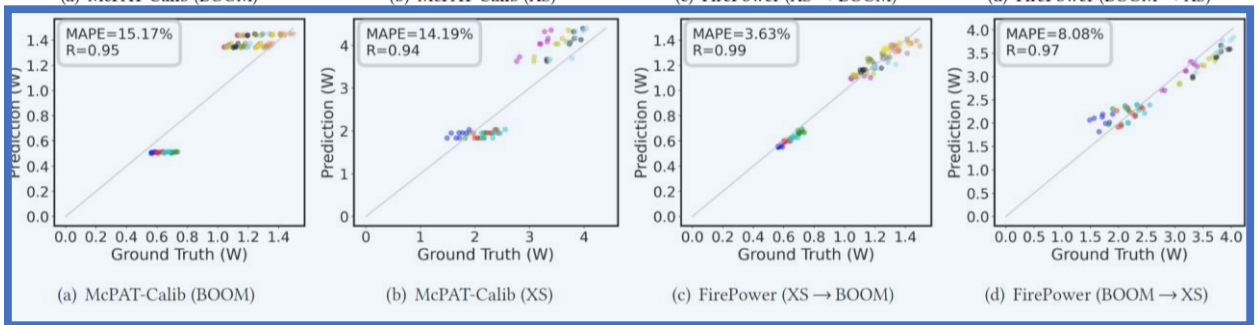
(a) McPAT-Calib (BOOM) (b) McPAT-Calib (XS) (c) FirePower (XS → BOOM) (d) FirePower (BOOM → XS)

Known 3 Config



(a) McPAT-Calib (BOOM) (b) McPAT-Calib (XS) (c) FirePower (XS → BOOM) (d) FirePower (BOOM → XS)

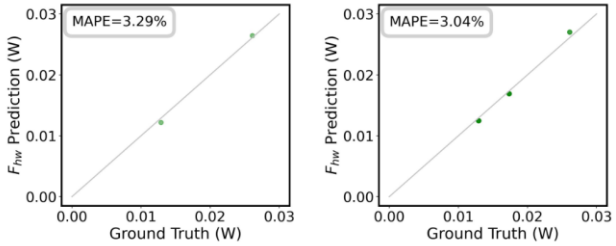
Known 2 Config



(a) McPAT-Calib (BOOM) (b) McPAT-Calib (XS) (c) FirePower (XS → BOOM) (d) FirePower (BOOM → XS)

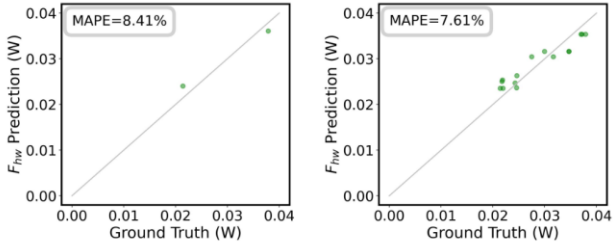
Only two configs as training data!

Generalization Evaluation Results



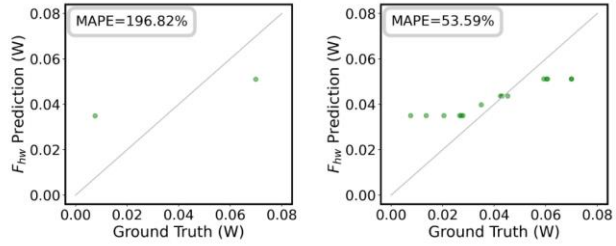
(a) D-TLB (Accessible Config of Target)

(b) D-TLB (All Config of Target)



(c) OtherLogic (Accessible Config of Target)

(d) OtherLogic (All Config of Target)

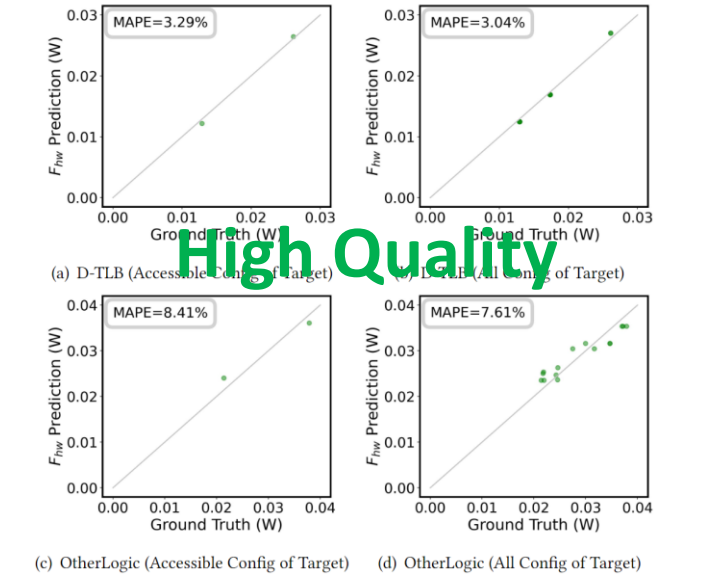


(a) LSU (Accessible Config of Target)

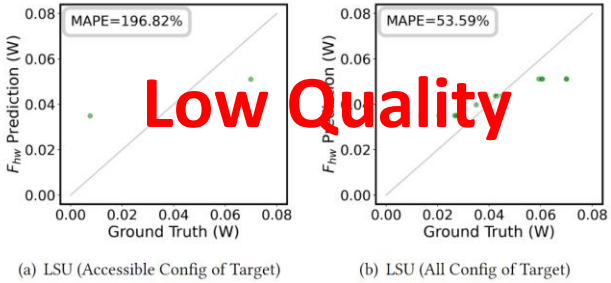
(b) LSU (All Config of Target)

- Comparison between accuracy observed from *accessible configs* and that of *all configs*
- Y-axis:
  - the ***adjusted prediction*** of the hardware model trained on accessible configs.
  - Multiply prediction with an ideal scaling factor

Generalization Evaluation Results



High Quality



Low Quality

- If the generalization quality observed with the accessible configurations has high accuracy
  - a high-quality generalized power model.
- Conversely
  - a low-quality generalized power model.

Conclusion

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

- A new paradigm FirePower that targets the few-shot learning scenario for new target architectures by different users
- Extracts the generalizable knowledge from a well-developed known architecture
- Utilize the generalizable knowledge to conduct few-shot power modeling on new target architecture

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

THANK YOU!

Qijun Zhang, Mengming Li, Yao Lu, and Zhiyao Xie

Hong Kong University of Science and Technology

qzhangcs@connect.ust.hk, eezhiyao@ust.hk