McPAT-Calib: A Microarchitecture Power Modeling Framework for Modern CPUs

Jianwang Zhai[1], Chen Bai[2], Binwu Zhu[2], Yici Cai[1], Qiang Zhou[1], Bei Yu[2]
[1]Tsinghua University [2]The Chinese University of Hong Kong

***Abstract*—Energy efficiency has become the core issue of modern CPUs, and it is difficult for existing power models to balance speed, generality, and accuracy. This paper introduces McPAT-Calib, a microarchitecture power modeling framework, which combines McPAT with machine learning (ML) calibration methods. McPAT-Calib can quickly and accurately estimate the power of different benchmarks running on different CPU configurations, and provide an effective evaluation tool for the design of modern CPUs. First, McPAT-7nm is introduced to support the analytical power modeling for the 7nm technology node. Then, a wide range of modeling features are identified, and automatic feature selection and advanced regression methods are used to calibrate the McPAT-7nm modeling results, which greatly improves the generality and accuracy. Moreover, a sampling algorithm based on active learning (AL) is leveraged to effectively reduce the labeling cost.**

We use up to 15 configurations of 7nm RISC-V Berkeley Out-of-Order Machine (BOOM) along with 80 benchmarks to extensively evaluate the proposed framework. Compared with state-of-the-art microarchitecture power models, McPAT-Calib can reduce the mean absolute percentage error (MAPE) of shuffle-split cross-validation by 5.95%. More importantly, the MAPE is reduced by 6.14% and 3.64% for the evaluations of unknown CPU configurations and benchmarks, respectively. The AL sampling algorithm can reduce the demand of labeled samples by 50%, while the accuracy loss is only 0.44%.

I. Introduction

Power modeling of integrated circuits is a broad and lasting research topic [1]. With the slowdown of Moore's Law and the breakdown of Dennard scaling, power consumption has become the bottleneck of modern CPUs performance. In industry, it is necessary to conduct accurate power-performance tradeoff analysis at the early design stage to ensure excellent CPU designs. At the same time, the design space of modern CPUs is very large, and designers need to conduct extensive design space exploration (DSE) and optimization to meet increasingly stringent design requirements. When performing DSE, it is necessary to model the entire design space (*i.e.*, different design parameter configurations along with different benchmarks), which put forward higher requirements for the speed, generality, and accuracy of the power model. The cumbersome analysis flow and large-scale DSE make power modeling a great challenge for the design of modern CPUs.

Currently, the most accurate power analysis is done by commercial gate-level power tools (*e.g.*, PrimeTime PX), and the flow is shown by the dotted-line in Fig. 1. Designers feed gate netlist into simulation and power analysis tools, which usually takes hours or even days to complete. For early DSE, this will be an unbearable high cost.

This work is supported by the National Natural Science Foundation of China 61834002, and Research Grants Council of Hong Kong SAR CUHK14209420, CUHK14208021.

TABLE I Comparison of Different Power Models

Model	Level	Speed	Generality	Accuracy
PrimeTime PX	Gate	Low	High	High
GRANNITE [2]	Gate	Medium	Medium	High
PRIMAL [3]	RTL	Medium	Medium	High
TCAD'17 [6]	Runtime	High	Low	High
McPAT [4]	Arch	High	High	Low
McPAT-Calib	Arch	High	High	High

To accelerate power modeling, academia has proposed many modeling methods at different design stages. The comparison of different power models is shown in TABLE I. It is not hard to see that the existing models are difficult to perform well in terms of speed, generality, and accuracy. At the gate or RTL level, power models (*e.g.*, GRANNITE [2] and PRIMAL [3]) are built based on simulation traces using feature engineering and ML methods, which can achieve higher accuracy. However, these models require netlist design and simulation, which is extremely time-consuming. And most models are design-specific and cannot estimate unknown configurations well. So it is difficult to use them for different CPU designs. At a higher microarchitecture level, power models use design parameters and event statistics for power modeling, which have the fastest modeling speed and higher generality. However, these models are difficult to capture hardware details, resulting in low modeling accuracy. Analytical power models (*e.g.*, McPAT [4]) are general and directly usable, but the modeling accuracy is too low [5] to meet the design requirements of modern CPUs. PMC-based runtime power models (*e.g.*, TCAD'17 [6]) can also be used at the microarchitecture level through transformation [7], but they are design-specific and cannot perform DSE.

Among these power models, McPAT [4] has gained popularity due to its ease-of-use and readiness. Hierarchical modeling from a circuit level to high-level system architecture provides a fast and general power estimation. McPAT enables designers to modeling power using only microarchitecture configurations with event statistics, without RTL design and simulation. However, two shortcomings make it difficult for original McPAT to help the design of modern CPUs: 1) The modeling accuracy is very low due to its incompleteness and architectural disparities in model; 2) McPAT lacks support for advanced technology nodes (*e.g.*, 7nm FinFET technology).

In this work, we propose a novel framework called McPAT-

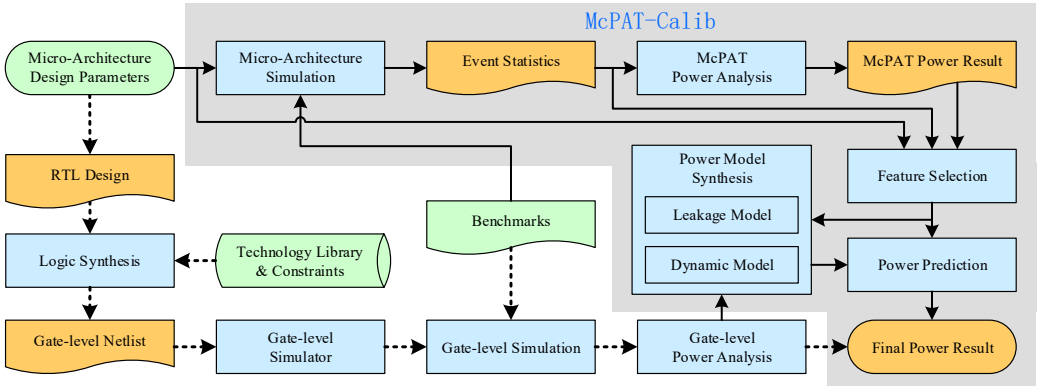


Fig. 1 Power Modeling Flow.

Calib, which combines internal improvements and ML calibration to McPAT to alleviate these problems. Firstly, internal improvements are made to obtain McPAT-7nm, which supports the analytical modeling of CPUs under 7nm FinFET technology. Secondly, automatic feature selection and advanced nonlinear regression are used to calibrate the McPAT-7nm modeling results, which greatly improves the accuracy while ensuring the generality of the power model. Moreover, a pre-clustering sequential AL sampling algorithm is leveraged to effectively reduce the labeling cost. The modeling flow of McPAT-Calib is shown by the solid-line in Fig. 1. It can quickly and accurately estimate the power of different benchmarks running on different CPU configurations, providing an effective tool for modern CPU design.

Our main contributions are summarized as follows:

- We obtain McPAT-7nm by introducing 7nm FinFET technology and microarchitecture modification. It is the first step to implement McPAT-Calib.
- We use advanced ML methods to calibrate McPAT-7nm, which can model dynamic and leakage separately. We identify a wide range of feature sources and overcome multicollinearity through automatic feature selection, to ensure the generality of the power model.
- We propose a pre-clustering sequential AL sampling algorithm, which selects the most useful samples to label, effectively reducing the labeling cost. It is the first application of AL in power modeling.
- We conduct an extensive evaluation of the proposed modeling framework, using up to 80 benchmarks along with 15 representative configurations of BOOM Core under 7nm FinFET technology. We also explore 15 regression models, among which the advanced nonlinear XGBoost Regressor performs best.

The rest of this paper is organized as follows. Section II introduces the important related work. Section III is an overview of the problem formation, modeling flow and RISC-V BOOM. Section IV provides detailed explanations on proposed McPAT-Calib. Section V conducts an extensive evaluation. Finally, Section VI concludes this paper.

II. Related Work

To avoid the high cost of gate-level power analysis, academia has proposed various modeling methods for early design stages. According to whether RTL design and simulation are needed, power modeling methods can be divided into two types: microarchitecture level, RTL (and lower) level.

At the microarchitecture level, power models only use limited design parameters and simulation information for modeling. Since there is no netlist design and simulation, the modeling speed is very fast. Analytical power models such as CACTI [8], Wattch [9], and McPAT [4] try to establish the internal hierarchical representation of CPUs and use event statistics to get the final estimate of power. McPAT is very influential, but its modeling error is as high as 20%-40% [5] [10] and lacks support for advanced technology nodes, which is difficult to meet the design requirements of modern CPUs. McPAT-PVT [11] and McPAT-Monolithic [12] realize technology node update and application range expansion, but the problem of insufficient accuracy is not considered. PowerTrain [10] uses regression calibration to improve the accuracy of McPAT, but the calibrated model is design-specific and cannot make good predictions for different configurations.

In addition to analytical models, some ML-based models can also be used for microarchitecture power modeling. Earlier works [13] [14] use design parameters to perform regression modeling to achieve DSE, but it lacks event statistics and cannot accurately model different benchmarks. The event-based [15] and PMC-based [6] [16] models can also be used. Jacobson H *et al.* [15] use a relatively small number of event statistics to perform power modeling. Walker MJ *et al.* [6] build runtime power model that allows separation of static and dynamic power by automatically selecting optimum PMC events. Sagi M *et al.* [16] use nonlinear transformation to capture relations between performance counters (PMC) and power, and use least-angle regression to complete the multivariate polynomial regression. Reddy BK *et al.* [7] transform the PMC-based empirical model [6] into microarchitecture power model. However, event statistics are closely related to CPU configuration, so these models are specific to CPU design, so it is difficult to perform DSE.

RTL and lower level power models require netlist design and simulation. PrEsto [17] uses linear model and feature engineering to characterize modules. Yang *et al.* [18] use a feature selection technique based on SVD to construct a single linear power model. PRIMAL [3] uses register

switching activities in simulation traces to build ML power models of reusable circuit building blocks. Simmani [19] uses VCD dumps to construct a toggle-pattern matrix, and selects key signals through clustering to build the power model. GRANNITE [2] represents the gate netlist as a graph, and takes register states and unit inputs from RTL simulation as features to construct the graph neural network (GNN) model to predict gate toggle rates and average power. These models can provide more hardware details, and often achieve cycle-by-cycle accurate modeling. However, these models have two fatal shortcomings: 1) Netlist design and simulation are needed, which is costly and slow; 2) Most models are design-specific and difficult to use for different CPU configurations.

III. Preliminaries

In this section, we will introduce the preliminary knowledge of microarchitecture power modeling.

A. RISC-V BOOM

RISC-V is an open-source instruction set architecture (ISA) that has received strong attention and support from academia and industry. Berkeley Out-of-Order Machine (BOOM) [20] utilizes Chisel [21] to construct a generator for the core, and is a family of out-of-order RISC-V designs rather than a single instance of a core. Due to the page limitation, more details about RISC-V BOOM can be found on the website [22].

Thanks to parametric microarchitecture design, designers can obtain different BOOM configuration by configuring the core with different design parameters, such as *FetchWidth*, *DecodeWidth*, *etc.* shown in TABLE III. The divergent trade-offs of BOOM between power and performance are needed to meet various design requirements. For a better BOOM design, a comprehensive DSE must be implemented, which requires accurate power modeling of different benchmarks running on different CPU configurations. Our modeling framework is dedicated to solving this problem to help the design of modern CPUs.

B. Problem Formulation

Generally, the total power P can be expressed as:

$$P = P_{dyn} + P_{leak} = \alpha C V_{DD}^2 f + V_{DD} I_{leakage} \quad (1)$$

where P_{dyn} is the dynamic power and depends on: the activity factor α, the switching capacitance C, the supply voltage V_{DD}, and the frequency f. The leakage power P_{leak} depends on V_{DD} and the leakage current $I_{leakage}$. However, modeling the CPU is very expensive at the transistor level, and we try to model power at a higher level of abstraction. At the same time, to meet the requirements of CPU design, it is necessary to accurately evaluate the power of different benchmarks running on the different target configurations. The microarchitecture power modeling problem can be defined as:

Problem 1 (Microarchitecture Power Modeling)**.** *Given different benchmarks $\mathcal{B}$ and different CPU configurations $\mathcal{C}$ characterized by microarchitecture design parameters. The objective of modeling is to estimate the power P_{ij} of each benchmark $B_j \in \mathcal{B}$ running on each configuration $C_i \in \mathcal{C}$.*

TABLE II Key Parameters of 7nm FinFET Technology

FinFET parameters	Value
Supply voltage, V_{DD} (V)	0.7
Gate length, L_G (nm)	21
Fin height, H_{FIN} (nm)	32
Fin thickness, T_{SI} (nm)	6.5
Fin pitch, F_P (nm)	27
Contacted poly-pitch, CPP (nm)	54

C. Power Modeling Flow

When the CPU configuration and benchmark are given, the traditional commercial power analysis flow is shown by the dotted-line in Fig. 1. RTL design and logic synthesis are required to obtain the gate-level netlist, and then gate-level simulation and power analysis are performed. The entire flow is extremely time-consuming. We use the commercial flow as a golden tool to obtain the ground truth of power.

The flow of our modeling framework is shown by the solid-line in Fig. 1, which can complete power modeling in a few seconds. gem5 [23] is used to complete the microarchitecture simulation of RISC-V BOOM to provide detailed event statistics for McPAT-Calib modeling. First, McPAT-7nm is used for preliminary analytical power modeling. Then, advanced ML calibration methods are used to obtain more accurate power results. Moreover, an AL sampling algorithm is used to reduce the demand for labeled samples during data acquisition.

IV. McPAT-Calib

A. Overview of McPAT-Calib

We give an overview of McPAT-Calib to make it easier to understand the proposed framework, including McPAT-7nm, ML Calibration, and AL Sampling.

Firstly, McPAT-7nm is used to complete the fast preliminary power modeling of CPUs. McPAT-7nm is obtained through internal improvements to McPAT, including the introduction of 7nm FinFET technology and microarchitecture modifications. Secondly, advanced ML methods are used to calibrate the McPAT-7nm results to obtain more accurate modeling results. We propose an automatic feature selection algorithm to overcome the multicollinearity of features to ensure the generality of the calibrated power model, and use an advanced nonlinear model to capture the nonlinearity in power modeling. Moreover, we propose a pre-clustering sequential AL sampling algorithm, which can effectively reduce the labeling cost in the data acquisition process.

B. McPAT-7nm

There are two ways to improve the accuracy of McPAT. The first way is internal improvement. The hierarchical analytical modeling method allows researchers to make internal modifications to provide more accurate power estimates, but this requires in-depth hardware analysis, which is very cumbersome. The second way is calibration, but the calibrated power model obtained by the existing method [10] is specific to the CPU design and has poor generality, making it difficult to provide a good estimate for unknown CPU configuration (*i.e.*, the CPU configuration with different design parameters).

To take advantage of the ease-of-use of the analytical power model, we introduce McPAT-7nm, which completes the preliminary power modeling of 7nm CPUs through simple internal improvements to McPAT. McPAT-7nm is the first part of the proposed framework and is the basis of McPAT-Calib. We first introduce 7nm technology parameters and reduce modeling errors through microarchitecture modification and empirical coefficient adjustment. These improvements are easy to implement and very effective.

Unlike ML-based methods, the analytical power model depends on specific technology parameters. The original McPAT only supports 180-22nm CMOS technology modeling. McPAT-PVT [11] and McPAT-Monolithic [12] updated it to 22nm and 14nm FinFET technology, but it is still difficult to meet the latest requirements of modern CPUs. Our target technology library is 7nm FinFET PDK ASAP7 [24], whose key parameters are shown in TABLE II. We obtain some parameters through scaling, and then get the parameters (*e.g.*, V/C/I) required for McPAT modeling, thus realizing McPAT's support for 7nm FinFET technology. In addition to technology parameters, McPAT also uses some empirical undifferetiated Core/FU coefficients. We adjust these empirical coefficients to reduce modeling errors in McPAT-7nm.

In original McPAT, the default pipeline is at least 12 stages, which is very different from RISC-V BOOM Core and will cause modeling errors. Therefore, we modify McPAT to support accurate modeling of the BOOM pipeline. In addition, the default minimum cache size exceeds some parameters of the low-configuration BOOM cores. Therefore, we modify it.

C. ML Calibration

Due to the high abstraction level of McPAT, it is difficult to capture all hardware details of modern CPUs, which inevitably brings low modeling accuracy. PowerTrain [10] only reweights and sums original McPAT results through linear regression to calibrate the total power, which leads to a lack of generality and accuracy.

We propose novel advanced ML methods to calibrate McPAT-7nm, which is the most important step of McPAT-Calib. To provide more power details, we choose to calibrate leakage and dynamic separately. To maintain the generality of McPAT-7nm, we chose appropriate modeling feature sources for leakage and dynamic power, and propose a feature selection method to automatically select the optimal features that reflect different CPU configurations and benchmarks. We use up to 15 regression models for evaluation, which proves the effectiveness of the proposed calibration methods.

1) Calibration Method and Feature Source: **Leakage calibration**. For one CPU configuration, its leakage is a fixed value (under a certain operating temperature). Therefore, we predict the $\overline{leakage}$ of different CPU configurations, *i.e.*, the average of all samples under the same configuration. This will cause the number of samples to be greatly reduced. So, we only select two useful features for leakage calibration, namely *Core.Leakage* and *Core.Area* obtained by McPAT-7nm.

Dynamic calibration. Dynamic is closely related to CPU configuration and benchmark. It is difficult to achieve high generality and accuracy by only using McPAT modeling results as features. As shown in Fig. 2, we choose to use McPAT-7nm results, microarchitecture design parameters, and event statistics as dynamic modeling features at the same time. For McPAT results, we use the dynamic of all levels of modules. *Core.Area* and *Core.Leakage* are also used, because they can provide useful information related to different configurations. It should be pointed out that gem5 can give thousands of event statistics. We selected 90 power-related features according to expert experience and divided them by *numCycles* for normalization. The correlation between each modeling feature and dynamic is shown in Fig. 2.

Algorithm 1 Filter Sequential Feature Selection

Require: $allFeatures$, all modeling features; k, the number of features to select; $varThreshold$, the variance threshold used to filter features;
Ensure: $selectedList$, the selected k optimal features;

```
for tmpFeature in allFeatures do
    if var(tmpFeature) ≤ varThreshold then;
        Delete tmpFeature from allFeatures;
    end if
end for
selectedList = ϕ;
while selectedList.length < k do
    bestR² = −inf;
    for tmpFeature in allFeatures do
        Cross-Validation(selectedList + tmpFeature);
        if newR² > bestR² then;
            bestR² = newR²;
            bestFeature = tmpFeature;
        end if
    end for
    Add bestFeature to selectedList;
    Delete bestFeature from allFeatures;
end while
```

2) Automatic Feature Selection: An important consideration when performing regression is multicollinearity (*i.e.*, the correlation between modeling features). A large number of features with high multicollinearity will lead to high model complexity and lack of stability, and severely lead to overfitting. Importantly, it will cause the power model to fail to accurately predict unknown configurations and benchmarks. The variance inflation factor (VIF) can be used to quantify multicollinearity. To find the VIF of a feature, an ordinary least squares linear regression model can be build which predicts this feature using the others. Then we can get:

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

where R^2 is *coefficient of determination*, as defined in Equation (4). Generally speaking, a VIF over 10 indicates strong multicollinearity. As shown in Fig. 2, the VIF of

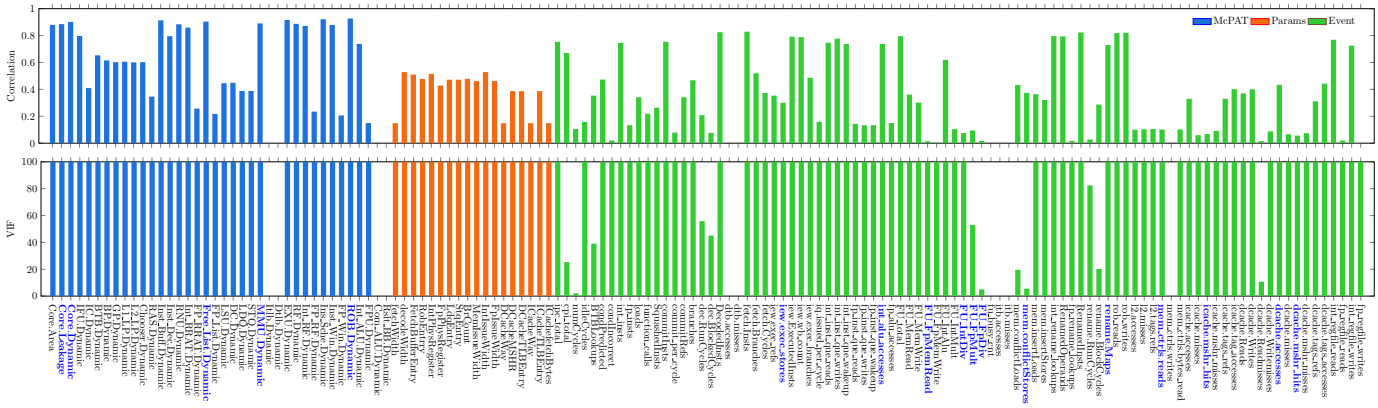


Fig. 2 Correlation with Dynamic and VIF of Dynamic Modeling Features.

most dynamic modeling features far exceeds this indicator, so the problem of multicollinearity cannot be ignored. We propose an automatic feature selection algorithm to solve this problem. Algorithm 1 first removes features with low variance (*i.e.*, filters out these features that have little impact on modeling) to reduce the search space, and then uses forward sequential selection to obtain the most beneficial k features. The effectiveness of feature selection is described in Section V-C.

3) Regression Model: Previous work [16] shows that there is strong nonlinearity in power modeling, which cannot be captured by linear models. To solve this problem, we use advanced nonlinear XGBoost Regressor (XGBR) as the final dynamic model. To evaluate the effectiveness of the proposed methods, we compare XGBR with 14 other regression models, including: (1) Linear Regression (LR); (2) Linear regression with L1 regularization (Lasso); (3) Linear regression with L2 regularization (Ridge); (4) Linear regression with L1 and L2 regularization (ElasticNet); (5) Bayesian Ridge Regression (BRR); (6) Gaussian Process Regression (GPR); (7) Regression based on K-Nearest Neighbors (KNNR); (8-9) Support Vector Regression with Polynomial (Poly_SVR) and Radial-Basis-Function (RBF_SVR) kernel; (10) Decision Tree Regressor (DTR); (11) Random Forest Regressor (RFR); (12) AdaBoost Regressor (ABR); (13) Gradient Boosting Regressor (GBR); (14) Bagging Regressor (BAGR).

D. Pre-clustering Sequential AL Sampling

To build an accurate power model, one needs to have a large number of labeled training samples, *i.e.*, samples whose modeling features and power ground truth are both known. Generally the more the labeled training samples are, the better the modeling performance is. However, labeling samples (*i.e.*, to get the ground truth of power) requires gate-level simulation and power analysis, which is time-consuming and an unacceptable cost. In our experiments, the whole flow for each sample takes approximately 5-20 hours.

Therefore, we propose a sampling algorithm based on active learning (AL) [25] to reduce the labeling cost in the training data acquisition process. It is the first application of AL in the field of power modeling. AL aims to use a well-designed algorithm to select the most useful training samples to label under a limited budget, to maximize learning performance at a lower labeling cost.

Contrary to obtaining the ground truth, it is easy to obtain modeling features of a sample and only takes a few seconds. We propose a pool-based sampling method, namely Pre-clustering Sequential AL Sampling, as shown in Algorithm 2. Suppose we have obtained the features of all N samples as an unlabeled sample pool $\{\boldsymbol{x}_n\}_{n=1}^{N}$. First, Algorithm 2 selects representative initial samples $\{\boldsymbol{x}_n\}_{n=1}^{d}$ to query label $\{y_n\}_{n=1}^{d}$ through clustering, so that the sample query strategy can build a basic model. Then, it enters the iterative stage, each time the most useful sample $\boldsymbol{x}$ is selected to query label y according to the sample query strategy, and the newly labeled sample $(\boldsymbol{x}, y)$ is added to the training set. Finally, these labeled samples $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{M}$ are used to build a model $f(\boldsymbol{x})$ that can achieve the best performance.

Algorithm 2 Pre-clustering Sequential AL Sampling

Require: $\mathcal{S}$, a set of unlabeled samples $\{\boldsymbol{x}_n\}_{n=1}^{N}$, where $\boldsymbol{x}_n \in \mathbb{R}^d$; M, the maximum number of samples to label;
Ensure: $\mathcal{K}$, the training set of labeled samples $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{M}$; $f(\boldsymbol{x})$, the regression model;
1: $\mathcal{K} = \phi$;
2: Perform k-means clustering on S to obtain d clusters, $\mathcal{C}_i, i = 1, ..., d$;
3: **for** $i = 1 : d$ **do**
4: Select the sample $\boldsymbol{x}$ closet to the center of $\mathcal{C}_i$ to label;
5: Add $(\boldsymbol{x}, y)$ to $\mathcal{K}$, delete $\boldsymbol{x}$ from $\mathcal{S}$;
6: **end for**
7: **for** $i = d + 1 : M$ **do**
8: Use the sample query strategy iGS to select the most beneficial sample $\boldsymbol{x}$ in $\mathcal{S}$ to label;
9: Add $(\boldsymbol{x}, y)$ to $\mathcal{K}$, delete $\boldsymbol{x}$ from $\mathcal{S}$;
10: **end for**
11: Use the training set $\mathcal{K}$ to build the power model $f(\boldsymbol{x})$.

1) Initial Samples Selection: Some existing AL methods [26] [27] usually select initial samples randomly, resulting in poor quality of initial samples. We use a better initialization

TABLE III Design Parameters and Power Statistics of Our 15 BOOM Configurations

Parameters	SmallBoomConfig			MediumBoomConfig			LargeBoomConfig			MegaBoomConfig			GigaBoomConfig		
	SE	Default	Pro	SE	Default	Pro	SE	Default	Pro	SE	Default	Pro	SE	Default	Pro
FetchWidth	4	4	4	4	4	8	8	8	8	8	8	8	8	8	8
DecodeWidth	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5
FetchBufferEntry	5	8	16	8	16	24	18	24	30	24	32	40	30	35	40
RobEntry	16	32	48	64	64	80	81	96	114	112	128	136	125	130	140
IntPhysRegister	36	52	68	64	80	88	88	100	112	108	128	136	108	128	140
FpPhysRegister	36	48	56	56	64	72	88	96	112	108	128	136	108	128	140
LDQ/STQEntriy	4	8	16	12	16	20	16	24	32	24	32	36	24	32	36
BranchCount	6	8	10	10	12	14	14	16	16	18	20	20	18	20	20
MemIssue/FpIssueWidth	1	1	1	1	1	1	1	1	2	1	2	2	2	2	2
IntIssueWidth	1	1	2	1	2	2	2	3	3	4	4	4	5	5	5
DCache/ICacheWay	2	4	8	4	4	8	8	8	8	8	8	8	8	8	8
DCache/ICacheTLBEntry	8	8	16	8	8	16	16	16	32	32	32	32	32	32	32
DCacheMSHR	2	2	4	2	2	4	4	4	4	4	4	8	8	8	8
ICacheFetchBytes	2	2	2	2	2	4	4	4	4	4	4	4	4	4	4
Min.Power(mW)	9.54	10.22	12.11	11.89	13.10	19.07	21.36	22.81	28.03	26.39	34.10	34.57	37.15	34.12	36.70
Max.Power(mW)	14.13	16.69	19.94	22.64	27.74	32.79	38.07	42.56	50.52	51.36	62.72	64.22	61.80	59.75	63.82
Avg.Power(mW)	11.76	13.53	15.64	16.42	17.94	24.60	28.02	30.02	35.97	36.55	44.06	45.52	45.62	43.26	46.38
Std.Power(mW)	1.22	1.70	1.73	2.81	3.95	3.76	4.62	5.00	5.56	6.06	7.27	7.84	6.00	6.64	7.10

approach, *i.e.* pre-clustering, to ensure the diversity and representativeness of initial samples. Assuming that there are N samples, we will select d initial samples, where d is the dimension of features. For N unlabeled samples, we perform k-means ($k = d$) clustering, and then select the sample closest to the cluster center from each cluster.

2) Sample Query Strategy: In each iteration, assuming that k samples $\{\boldsymbol{x}_m\}_{m=1}^{k}$ have already been labeled with outputs $\{y_m\}_{m=1}^{k}$, the sample query strategy is used to select the most useful sample from the remaining $N-k$ unlabeled samples $\{\boldsymbol{x}_n\}_{n=k+1}^{N}$ to label . We use an advanced query strategy iGS [28], which selects new samples to increase the diversity in both feature and label spaces. First, the feature distance d_{nm}^{x} between each unlabeled sample $\boldsymbol{x}_n$ and each labeled sample $\boldsymbol{x}_m$ is computed. Then, a power model $f(\boldsymbol{x})$ is built using the labeled samples, and the label distance d_{nm}^{y} between the prediction $f(\boldsymbol{x}_n)$ of each unlabeled sample $\boldsymbol{x}_n$ and the label y_m of each labeled sample $\boldsymbol{x}_m$ is computed. Finally, iGS computes $d_n^{xy} = \min_m d_{nm}^{x} d_{nm}^{y}$ of each unlabeled sample and selects the sample with the maximum d_n^{xy} to label.

3) Stop Criteria: Since sample labeling is extremely time-consuming, we can only label a limited number of samples under a given budget. When the maximum number M is reached, the sampling algorithm stops and uses all selected labeled samples to build the final power model $f(\boldsymbol{x})$.

V. Evaluation

We conduct a comprehensive evaluation of proposed framework. We first introduce the experimental settings in Section V-A, and then evaluate the effectiveness of proposed methods in Section V-B to Section V-E. More importantly, we compare McPAT-Calib with previous work to prove its superiority, as shown in Section V-F.

A. Experiments Settings

We have expanded the official 5 RISC-V BOOM configurations, and obtain a total of 15 configurations for evaluation. Their design parameters are evenly distributed in the design space, and they are quite different from each other, enough to evaluate the generality of power models. Design parameters and power statistics are shown in TABLE III.

To comprehensively evaluate the power performance of CPUs under different workloads, we select up to 80 benchmarks as shown in Fig. 5, covering commonly used benchmarks. The sources are as follows: 8 benchmarks from *riscv-tests* [29]; 19 representative isa from *riscv-tests* by executing 200 loops to increase the ratio of effective instructions to prevent incorrect estimates; most of the rest 53 benchmarks are derived from previous open source projects [30] [31]. We make simple modifications to some benchmarks to enable them to complete RISC-V simulations.

We use 7nm ASAP7 PDK [24] and commercial gate-level power analysis flow (Genus for logic synthesis, VCS for simulation @ 500MHz, and PrimeTime PX for power analysis) to obtain the power ground truth. All 1200 samples are obtained for evaluation, using 15 BOOM configurations along with 80 benchmarks.

Mean absolute percentage error (MAPE) and *coefficient of determination* (R^2) are two metrics to evaluate the accuracy of power modeling results, which are defined as:

$$\text{MAPE} = \frac{1}{n}\sum_{i}^{n}\frac{|p_i^{pred} - p_i^{truth}|}{p_i^{truth}} \times 100\%, \tag{3}$$

$$R^2 = 1 - \frac{\sum_i^n (p_i^{pred} - p_i^{truth})^2}{\sum_i^n (p_i^{truth} - \overline{p})^2}, \tag{4}$$

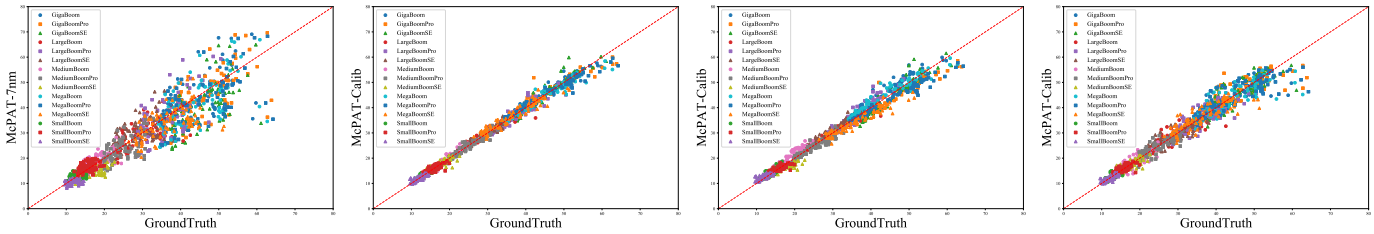
where p^{truth} is the power truth, p^{pred} is the power estimation, and $\overline{p} = \frac{1}{n}\sum_i^n p_i^{truth}$ is the average of power truth.

B. McPAT-7nm

As an analytical power model, McPAT-7nm can be used directly without training. Therefore, it can directly model the power of all samples. The scatter diagram of McPAT-7nm modeling results and ground truth is shown in Fig. 3(a), where MAPE $= 13.02\%$ and $R^2 = 0.817$.

C. McPAT-Calib: Leakage and Dynamic Power

Unlike analytical power models, data-driven ML methods need to use labeled samples to train a model and then make predictions. Therefore, we use cross-validation [32] to evaluate the 15 regression models described in Section IV-C3.



(a) McPAT-7nm (b) McPAT-Calib: Shuffle-Split (c) McPAT-Calib: Unknown Config. (d) McPAT-Calib: Unknown Bench.

Fig. 3 Power Modeling Results vs. Ground Truth.

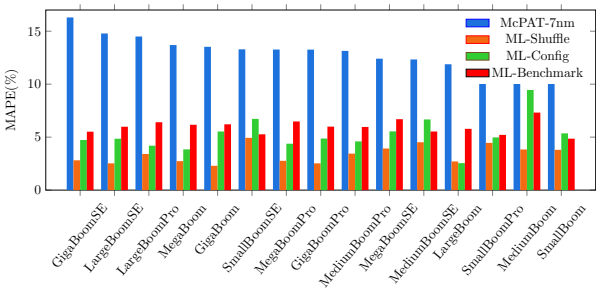


Fig. 4 Power Modeling Results of Different Configurations.

1) Leakage Power: Leakage is bound to the CPU configuration. Therefore, we mainly evaluate the predictive ability of leakage for unknown CPU configurations. Specifically, the average $\overline{leakage}$ ground truth under each CPU configuration is taken as the prediction target, so there are 15 samples in total. We use 14 samples representing 14 configurations to train the leakage model and predict another unknown configuration, and repeat 15 times. The results of each model are shown in TABLE IV, where the 2-degree Ploy_SVR can achieve the best modeling results with MAPE $= 4.47\%$.

2) Dynamic Power: To evaluate the generality of the power model, we perform 15-fold cross-validation [32] according to configuration, train the dynamic model with samples in 14 known configurations, and predict the dynamic of samples in another unknown configuration. More importantly, we compared the modeling results using all dynamic modeling features and the selected features after automatic feature selection. The results show that all models can get better results after feature selection, which can effectively avoid overfitting and improve the generality of the model.

Dynamic-Total Features. We first use all features for modeling, and the results are shown in TABLE IV. The advanced non-linear model XGBR achieves the best result with MAPE $= 7.40\%$, and MAPE of most linear regressors are up to 15-20%. It can be seen that when all features are used for dynamic modeling, the results are not ideal. This is caused by the multicollinearity of features. The model is overfitted on the training samples of known configurations and cannot predict the samples of unknown configurations well.

Dynamic-Selected Features. As shown in TABLE IV, after automatic feature selection, fewer modeling features are used to get better modeling results. The modeling accuracy of all regressors has been improved, especially for linear models. This verifies the effectiveness of the proposed feature selection algorithm, which effectively overcomes the adverse effects of multicollinearity and automatically selects modeling features that can reflect different configurations and benchmarks. After feature selection, the best modeling result is still achieved by XGBR, where MAPE $= 6.23\%$ using 17 automatically selected features (with blue fonts in Fig. 2).

TABLE IV Leakage and Dynamic Modeling Results

Regressors	$\overline{Leakage}$	Dynamic-Total F.		Dynamic-Selected F.		
	MAPE	MAPE	R^2	k^*	MAPE	R^2
LR	7.34%	20.85%	0.816	48	7.40%	0.954
Lasso	8.08%	17.97%	0.869	48	7.55%	0.951
Ridge	7.10%	21.88%	0.790	48	7.31%	0.954
ElasticNet	6.77%	16.36%	0.889	22	9.20%	0.929
BRR	7.74%	18.50%	0.867	48	7.30%	0.954
GPR	7.72%	16.29%	0.895	15	9.32%	0.924
KNNR	8.21%	20.64%	0.783	13	13.21%	0.903
Poly_SVR	**4.47%**	35.04%	0.462	18	9.34%	0.923
RBF_SVR	6.09%	31.41%	0.504	21	8.99%	0.940
DTR	7.76%	14.70%	0.877	22	11.61%	0.914
RFR	7.46%	10.56%	0.943	6	8.09%	0.958
ABR	7.64%	14.24%	0.907	11	13.26%	0.893
GBR	8.88%	10.98%	0.936	28	9.25%	0.943
BAGR	7.59%	11.41%	0.931	6	9.92%	0.933
XGBR	7.81%	**7.40%**	**0.961**	17	**6.23%**	**0.969**

* k: The Number of Selected Features.

D. McPAT-Calib: Total Power

In this subsection, we give the total power results. More critically, the generality (*i.e.*, the ability to model unknown CPU configurations and benchmarks) of the proposed framework is evaluated. According to the leakage and dynamic modeling results, we use Ploy_SVR to build the leakage model and use XGBR to build the dynamic model with 17 selected features. The sum of the outputs of the two models is the modeling result of total power. Unless otherwise specified, all powers in the following refer to the total power, and McPAT-Calib's power model is implemented in this way.

1) Shuffle-Split Cross-Validation (ML-Shuffle): We first perform 15-fold shuffle-split cross-validation on all samples, and the scatter diagram of the modeling results is shown in Fig. 3(b), where MAPE $= 3.38\%$, $R^2 = 0.989$.

2) Modeling Unknown Configurations (ML-Config): To determine whether a power model can help the DSE of modern CPUs, the most important thing is to evaluate whether it can accurately model unknown CPU configurations, *i.e.*, configurations with different design parameters. Therefore, we use 1 previously invisible configuration as the test set and use the remaining samples with 14 known configurations

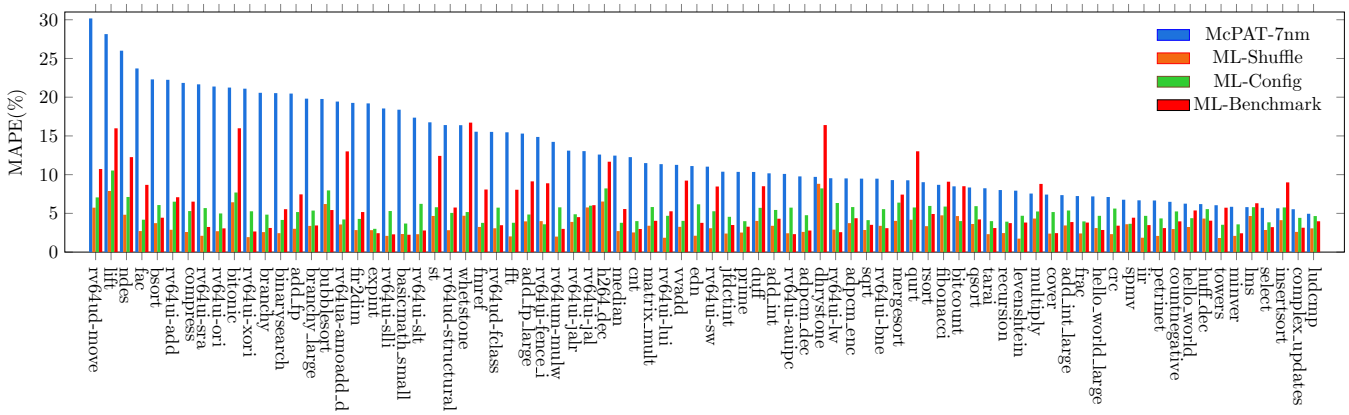


Fig. 5 Power modeling results of different benchmarks.

as the training set to build the calibrated power model. A 15-fold cross-validation is performed, and the modeling results are shown in Fig. 3(c), where MAPE $= 5.22\%$, $R^2 = 0.978$.

3) Modeling Unknown Benchmarks (ML-Benchmark): Similar to unknown configurations, the power model's ability to model unknown benchmarks is also a manifestation of its generality. To model unknown benchmarks, 4 previously invisible benchmarks are used as the test set, and the remaining benchmarks were used as the training set. A 20-fold cross-validation is performed, and the modeling results are shown in Fig. 3(d), where MAPE $= 5.96\%$, $R^2 = 0.958$.

Fig. 4 and Fig. 5 show the modeling accuracy of different CPU configurations and benchmarks. As shown in the figures, under different evaluation strategies, the modeling accuracy of McPAT-Calib is much higher than McPAT-7nm, which proves the accuracy and generality of the ML calibration methods.

E. AL Sampling

The AL sampling algorithm is added to the modeling for unknown configurations. Specifically, it performs 15-fold cross-validation according to CPU configurations. In each validation process, the AL sampling algorithm is used to select a specific number of beneficial samples from 1120 unlabeled training samples to label, and a calibrated power model is built to predict 80 test samples. To eliminate randomness, we take the average of 15 validation processes. TABLE V shows the test MAPE under several typical sampling ratios.

The result shows that our pre-clustering sequential AL sampling algorithm can effectively reduce the demand for labeled samples, and high modeling accuracy can be achieved when only a limited number of labeled samples are used. Our AL sampling algorithm can reduce the demand for labeled samples by 50% with only a 0.44% loss of accuracy. When the labeled samples reach a certain ratio (*e.g.*, 60%), the modeling error tends to converge, showing smaller fluctuations.

F. Comparison with Previous Work

To prove the superiority of our modeling framework, McPAT-Calib is compared with previous microarchitecture power modeling methods. As described in Section II, it is compared with the parameters-driven model HPCA'07 [14], the event-driven modeling methods TCAD'17 [6] and TCAD'20 [16], and the McPAT-based calibration method PowerTrain [10]. As shown in TABLE VI, McPAT-Calib is much higher than previous work in terms of accuracy and generality. Compared with state-of-the-art models, McPAT-Calib can reduce the MAPE of shuffle-split cross-validation by 5.95%; more importantly, it reduces the MAPE of estimating unknown CPU configurations by 6.14%, and the MAPE of estimating unknown benchmarks by 3.64%.

TABLE V MAPE under several typical sampling ratios

Ratio	10%(112)	20%(224)	30%(336)	40%(448)	50%(560)
MAPE	8.68%	6.91%	6.41%	5.92%	5.66%
Ratio	60%(672)	70%(784)	80%(896)	90%(1008)	100%(1120)
MAPE	5.65%	5.77%	5.47%	5.56%	5.22%

TABLE VI Comparison with previous work

Methods	Shuffle-Split		Unknown Config.		Unknown Bench.	
	MAPE	R^2	MAPE	R^2	MAPE	R^2
HPCA'07 [14]	15.31%	0.807	18.37%	0.752	15.34%	0.807
TCAD'17 [6]	11.71%	0.899	14.31%	0.875	13.56%	0.842
TCAD'20 [16]	22.51%	0.746	24.58%	0.711	23.92%	0.690
PowerTrain [10]	9.33%	0.926	11.36%	0.906	9.60%	0.921
McPAT-7nm	13.02%	0.817	13.02%	0.817	13.02%	0.817
McPAT-Calib	**3.38%**	**0.989**	**5.22%**	**0.978**	**5.96%**	**0.958**

VI. Conclusion

We propose a microarchitecture power modeling framework named McPAT-Calib that combines McPAT-7nm and advanced ML calibration methods, taking into account generality and accuracy. First, we made internal improvements to obtain McPAT-7nm to support power modeling of 7nm CPUs; Then, the ML calibration methods such as automatic feature selection and advanced nonlinear regression are used to improve the accuracy of McPAT-7nm; Finally, to reduce the labeling cost of ML calibration, a sampling method based on active learning is proposed, which effectively reduces the demand for labeled samples. We have extensively evaluated the proposed framework, and its superiority is proved by experimental results. This work will effectively promote the modeling and design of modern CPUs.

References

[1] Y. Nasser, J. Lorandel, J. C. Prévotet, and M. Hélard, "RTL to transistor level power modeling and estimation techniques for FPGA and ASIC: A survey," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2021, pp. 479–493.

[2] Y. Zhang, H. Ren, and B. Khailany, "GRANNITE: Graph neural network inference for transferable power estimation," in *ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[3] Y. Zhou, H. Ren, Y. Zhang, B. Keller, B. Khailany, and Z. Zhang, "PRIMAL: power inference using machine learning," in *ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6.

[4] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2009, pp. 469–480.

[5] S. L. Xi, H. Jacobson, P. Bose, G. Y. Wei, and D. Brooks, "Quantifying sources of error in McPAT and potential impacts on architectural studies," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2015, pp. 577–589.

[6] M. J. Walker, S. Diestelhorst, A. Hansson, A. K. Das, S. Yang, B. M. Al-Hashimi, and G. V. Merrett, "Accurate and stable run-time power modeling for mobile and embedded CPUs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, pp. 106–119, 2017.

[7] B. K. Reddy, M. J. Walker, D. Balsamo, S. Diestelhorst, B. M. Al-Hashimi, and G. V. Merrett, "Empirical CPU power modelling and estimation in the gem5 simulator," in *IEEE International Workshop on Power And Timing Modeling, Optimization and Simulation (PATMOS)*, 2017, pp. 1–8.

[8] S. Li, K. Chen, J. H. Ahn, J. B. Brockman, and N. P. Jouppi, "CACTI-P: Architecture-level modeling for SRAM-based structures with advanced leakage reduction techniques," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2011, pp. 694–701.

[9] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: a framework for architectural-level power analysis and optimizations," in *IEEE/ACM International Symposium on Computer Architecture (ISCA)*, 2000, pp. 83–94.

[10] W. Lee, Y. Kim, J. H. Ryoo, D. Sunwoo, A. Gerstlauer, and L. K. John, "PowerTrain: A learning-based calibration of McPAT power models," in *IEEE International Symposium on Low Power Electronics and Design (ISLPED)*, 2015, pp. 189–194.

[11] A. Tang, Y. Yang, C. Lee, and N. K. Jha, "McPAT-PVT: Delay and power modeling framework for FinFET processor architectures under PVT variations," *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, pp. 1616–1627, 2015.

[12] A. Guler and N. K. Jha, "McPAT-Monolithic: An area/power/timing architecture modeling framework for 3-d hybrid monolithic multicore systems," *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, pp. 2146–2156, 2020.

[13] B. C. Lee and D. M. Brooks, "Accurate and efficient regression modeling for microarchitectural performance and power prediction," in *ACM SIGOPS operating systems review*, 2006, pp. 185–194.

[14] ——, "Illustrative design space studies with microarchitectural regression models," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2007, pp. 340–351.

[15] H. Jacobson, A. Buyuktosunoglu, P. Bose, E. Acar, and R. Eickemeyer, "Abstraction and microarchitecture scaling in early-stage power modeling," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2011, pp. 394–405.

[16] M. Sagi, N. A. V. Doan, M. Rapp, T. Wild, J. Henkel, and A. Herkersdorf, "A lightweight nonlinear methodology to accurately model multicore processor power," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2020, pp. 3152–3164.

[17] D. Sunwoo, G. Y. Wu, N. A. Patil, and D. Chiou, "PrEsto: An FPGA-accelerated power estimation methodology for complex systems," in *International Conference on Field Programmable Logic and Applications*, 2010, pp. 310–317.

[18] Jianlei Yang, Liwei Ma, Kang Zhao, Yici Cai, and Tin-Fook Ngai, "Early stage real-time SoC power estimation using RTL instrumentation," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2015, pp. 779–784.

[19] D. Kim, J. Zhao, J. Bachrach, and K. Asanović, "Simmani: Runtime power modeling for arbitrary rtl with automatic signal selection," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2019, pp. 1050–1062.

[20] K. Asanovic, D. A. Patterson, and C. Celio, "The berkeley out-of-order machine (BOOM): An industry-competitive, synthesizable, parameterized RISC-V processor," University of California at Berkeley Berkeley United States, Tech. Rep., 2015.

[21] J. Bachrach, H. Vo, B. Richards, Y. Lee, A. Waterman, R. Avižienis, J. Wawrzynek, and K. Asanović, "Chisel: constructing hardware in a scala embedded language," in *ACM/IEEE Design Automation Conference (DAC)*, 2012, pp. 1212–1221.

[22] https://boom-core.org.

[23] A. Roelke and M. R. Stan, "RISC5: Implementing the RISC-V ISA in gem5," in *First Workshop on Computer Architecture Research with RISC-V (CARRV)*, 2017.

[24] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthy, and G. Yeric, "ASAP7: A 7-nm finFET predictive process design kit," in *Microelectronics Journal*, 2016, pp. 105–115.

[25] B. Settles, "Active learning literature survey," 2009.

[26] T. RayChaudhuri and L. Hamey, "Minimisation of data collection by active learning," in *International Conference on Neural Networks (ICNN)*, 1995, pp. 1338–1341.

[27] W. Cai, Y. Zhang, and J. Zhou, "Maximizing expected model change for active learning in regression," in *IEEE International Conference on Data Mining (ICDM)*, 2013, pp. 51–60.

[28] D. Wu, C.-T. Lin, and J. Huang, "Active learning for regression using greedy sampling," *Information Sciences*, pp. 90–105, 2019.

[29] https://github.com/riscv/riscv-tests.

[30] https://github.com/mcmenaminadrian/taclebench-riscv-baremetal.

[31] J. Pallister, S. J. Hollis, and J. Bennett, "Beebs: Open benchmarks for energy measurements on embedded platforms," *arXiv preprint arXiv:1308.5174*, 2013.

[32] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *International Joint Conference on Artificial Intelligence, (IJCAI)*, 1995, pp. 1137–1143.