Early Stage Real-Time SoC Power Estimation Using RTL Instrumentation*

Jianlei Yang[1,2], Liwei Ma[2], Kang Zhao[2], Yici Cai[1] and Tin-Fook Ngai[2]

[1]Department of Computer Science and Technology, Tsinghua University, 100084, Beijing, China

jerryyangs@gmail.com, caiyc@mail.tsinghua.edu.cn

[2]Intel Labs China, Intel Corporation, 100190, Beijing, China

{liwei.ma, kang.zhao@intel.com, tin-fook.ngai}@intel.com

ABSTRACT

Early stage power estimation is critical for SoC architecture exploration and validation in modern VLSI design, but real-time, long time interval and accurate estimation is still challenging for system-level estimation and software/hardware tuning. This work proposes a model abstraction approach for real-time power estimation in the manner of machine learning. The singular value decomposition (SVD) technique is exploited to abstract the principle components of relationship between register toggling profile and accurate power waveform. The abstracted power model is automatically instrumented to RTL implementation and synthesized into FPGA platform for real-time power estimation by instrumenting the register toggling profile. The prototype implementation on three IP cores predicts the cycle-by-cycle power dissipation within 5% accuracy loss compared with a commercial power estimation tool.

Categories and Subject Descriptors

B.8.2 [**Integrated Circuits**]: Performance and Reliability—*Performance Analysis and Design Aids*

General Terms

Algorithms, Design, Performance, Measurement

Keywords

Real-Time, Power Estimation, RTL Instrumentation, Singular Value Decomposition (SVD)

1. INTRODUCTION

Power consumption has become one of biggest challenges for modern chip design. In fact, power dissipation is regarded as a likely limiting factor to the increasing scales of integration predicted by Moore's Law [1]. Early visibility into power budgeting requires appropriate tools support for power estimation and optimization at various design stages.

Extensive research has addressed the power consumption issues at varying levels of abstraction. At lower level of design hierarchy, higher accuracy of analysis can be achieved because more detail on circuit implementation is available. These kinds of technologies have been incorporated into various commercial power estimation tools, such as Synopsys PrimeTime PX [2][3][4][5]. Higher level approaches usually perform functional simulation to calculate the power consumption and subsequently have larger capacity in terms of both transistor count and simulation time [6][7][8]. As the advances in fabrication technologies have led to shrinking devices sizes, and consequently increasing chip complexities, the larger scale circuit design and verifications become increasingly difficult and time consuming. The poor speed of power estimation tools limits their utility in the design flow. Clearly, such estimation tools cannot be used in an iterative manner for architectural exploration. Raising the level of abstraction to the architecture level can lead to substantial efficiency improvements, and many types of virtual platform technologies are proposed and renowned for early development and validation for the chip design. Especially, the accelerated hardware/software co-emulation is the most popular virtual platform to validate both functionality and performance which is essential for shorting turn-around time.

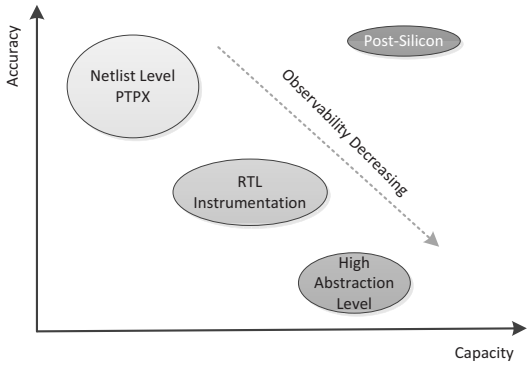

Figure 1: Solution space of power estimation methods

However, raising the abstraction level is not universal, which will bring obvious decreasing estimation accuracy. As shown in Figure 1, netlist power estimation methods have high accuracy and excellent observability. But its biggest drawback is their limited capacity. For post-silicon front, it has a perfect accuracy and can run long workloads. But post-silicon measurements come too late to influence the SoC architecture and consequently lead to limited observability.

Many interesting techniques have been developed for doing early estimations with reasonable accuracy [9][10][11][12]. In [11], a micro-architectural power model fed by activity counters is programmed into the chip multiprocessors to explore performance, power, and thermal issues. In [10], a

*This work was supported by the National Natural Science Foundation of China (NSFC) under Grant No.61274031 and No.61106030.

978-1-4799-7792-5/15/$31.00 ©2015 IEEE

power emulation approach is proposed for accelerating power estimation by mapping the resulting power model enhanced circuit onto a hardware prototyping platform. In [9], an FPGA-accelerated power estimator *PrEsto* is proposed with linear power models and integrated into an FPGA-based performance simulators of microprocessors. Despite these available approaches, accurate real-time power model is still a critical challenge for such approach while the increasing in number and complexity of the system has exerted tremendous pressure. Some of these approaches are mainly based on linear or parametric regression models to exploit the correlation between power consumption and input-output switching activities of different circuit components. But they are required to manually or incrementally select the candidate signals or state traces which are not optimal enough for minimizing the estimation errors.

Furthermore, real-time power profiling requirements cannot be satisfied by the approaches above. For current commercial tools such as PrimeTime PX, they have high accuracy and observability, but they are mostly post-processing tools which cannot support real-time power estimation. In addition, their estimation speed is very low for high-complex circuits or systems; sampling hundreds of signals in emulation has an overhead of extra routing and IO. Therefore, real-time power estimation based on a synthesized model is necessary. At a higher abstraction level, designers often need to verify the target system without hardware details in advance, which requires a real-time and fast estimation with a relative accuracy. It is just the motivation of our research. We will use the RTL instrumentation technology to satisfy the real-time requirements.

To overcome these limitations above, we introduce a RTL instrumentation technique for real-time power estimation in a machine learning manner. As shown in Figure 1, the RTL instrumentation has relative high capacity and observability with little accuracy loss. A concept of model abstraction is exploited for power characterization according to the circuit toggling profile by *singular value decomposition* (SVD) approach. The model is driven by *critical registers* instead of input-output signals like [9]. The introduced SVD approach automatically captures the relationship between the critical registers toggling profile and obtained power trace. This machine learning strategy avoid manually or incrementally selecting the decision variables since it could measure the influence of each toggling register on the total power consumption. Finally the abstracted power model is integrated/instrumented into RTL and synthesized onto a hardware prototyping platform with a small amount of additional logic resources. Experimental results with the design of H.264/AVC, AES and AC97 IP cores show that the proposed approach is extremely efficient and accurate for power calibration and real-time prediction.

The rest of the paper is organized as follows. The general methodology and our motivation are introduced in Section 2. The model abstraction and detailed implementations are presented in Section 3. Experimental results of power trace calibration and prediction are shown in Section 4. Concluding remarks are provided in Section 5.

2. PRELIMINARIES

2.1 General Methodology

Power consumption can be accurately calculated using simulation approach when detailed circuit information is available, but extremely time-consuming and consequently impractical for large scale designs. And such method cannot satisfy the real-time requirement. Ideally, we would like to have power models that can essentially characterize the relationship between power consumption and certain important circuit observations, such as circuit activities. The resulting power model is expected to be simple and easily implemented with RTL but still relative accurate. The challenge of the abstraction task lies in reducing the complexity of detailed power models without sacrificing accuracy. According to the observations from circuit simulation and analysis [13], power dissipation has a strong correlationship to certain components or ports. This suggests that the total power dispassion could be represented using a small number of key contributor items. A primary motivation is to select part of them as observation points and monitor their activities for representing their total power dispassion.

The power estimation generally includes two phases: modeling/training phase and prediction phase. On the modeling phase, part of the circuit activities is observed and power consumption is calculated by simulation on a certain number of cycles. In order to capture their essential relationship a popular way is to perform regression analysis between circuit activities and power traces [8][9][10][11][12][13][14][15]. Consequently, each coefficient for each observation point could be obtained to represent its contribution to the total power consumption, i.e., the sensitivity of the total power dissipation to each observation point. On the prediction phase, the above coefficients and the corresponding power model are exploited to calculate the power consumption in the future cycles. If the coefficients are applied on the referred training cycles, this kind of measurement is referred as *power trace calibration*. Or they are applied on the future cycles, it is referred as *power trace prediction*.

2.2 Research Motivation

The primary difference among the existing approaches lies in how to choose observation points and how to perform training on them. Most of them generally adopt the boundary information such as input/output signals as observation points supposing that the design has been divided into certain number of modules [9][12]. This strategy requires certain understanding from the structure/architectures in the design, and consequently it is not easy applicable for different designs. Aiming to reduce the complexity of the power model in [9], an incremental approach is adopted to select the candidate signals or state traces so that fewer signals and limited terms are kept in the linear power model. Meanwhile, the Hamming distances from cycle-to-cycle are further employed to reduce the number of bits for data buses instead of using individual bits. However, the work in [13] has explored the difference between power dissipation associated with different input transitions, as a function of the Hamming distance, between the corresponding transitions vectors, and indicated that the correlation between the two quantities is weak. Thus, the degree of optimization should be adjusted arbitrarily according to the complexity of the design and the desired accuracy.

The motivation of our approach is originally arisen from an important observation, i.e., the power dissipation of a circuit is more sensitive to some *primary registers* than to some input or output signals. After all, the abstraction level of registers is more close to logic/transistors level while the abstraction level of input/output signals on the module boundary cannot completely reflect the influences of many internal terms on total power dissipation in some cases. And from the point view of logic circuit level, a register is essentially a typical loading for previous stage, and has several fan-outs to drive. The register terms are more preferred to be chosen as observation points for representing the key contributor

items on the total power dissipation. The dynamic power consumption for a circuit is typically given as

$$P_{dyn} = \frac{1}{2}\alpha f C_{eff} V^2 \quad (1)$$

where α is a proportionality constant of transition probability, V is the power supply voltage, f is the operating frequency, and C_{eff} is the effective capacitance. The voltage and frequency for all circuits will be constant in a given design. In some cases, the switching activity and capacitance are combined as *switching capacitance* and the utilization factor of resources is considered as a separate entity in itself. Of the two remaining variables, switching activity is expected to be accounted for by register toggles, while the capacitance by the resource utilization which can be regarded as the effective capacitance of each register. For register i, define its toggle state as s_i (derived from α and f) and equivalent coefficient as β_i (derived from C_{eff} and V), then the power consumption can be written as

$$P_{dyn} \triangleq \sum_{i=1}^{N} s_i \beta_i. \quad (2)$$

for a circuit with N registers. Power model abstraction is to select a subset n from N registers as observation points and predictor variables, the power consumption P_{est} as the response variable, the relationship between the response and the predictor variables can be modeled as a linear equation as follows

$$P_{est} = s_1\beta_1 + s_2\beta_2 + \cdots + s_i\beta_i + \cdots + s_n\beta_n \quad (3)$$

where s_i is the transition state of each register, and β_i is the coefficient of each register, capturing the constant portion of the dynamic power for representing the contribution to the power consumption.

Model abstraction is the intelligent capture of the essence of the behavior of a model, without all the implementation details. Generally speaking, an abstracted model has nearly the same accuracy of its detailed counterpart, while having its complexity reduced to more closely match other components within different run scenarios. In machine learning community, singular value decomposition is known as one of ideal approaches to perform data dimensionality reduction for principal component analysis, and subsequently is suitable for finishing the model abstraction tasks. Consequently, SVD method is explored in this work to automatically capture the relationship between the critical registers toggling profile and the obtained power trace.

3. POWER MODEL ABSTRACTION

3.1 Framework

The proposed estimation approach is originally arisen from an important observation, that is, the power consumption trace is strongly correlated to the registers toggling profile especially for some critical registers. As shown in Figure 2, for a certain amount of cycles as input training data, the register toggling profile is dumped as $\mathbf{X}$, and the power consumption is calculated as P_{real} by gate level power analysis tools. Aiming to abstract the potential relationship between $\mathbf{X}$ and P_{real}, we should determine the coefficients $\boldsymbol{\beta}$ so that the estimated power consumption P_{est} is approximately equal to P_{real}. In general, the decision variables s_i in equation (3) are not independent due to the circuit coupling with each other. Even though the power model is proposed as linear formulation, the actual dependence of each power consumption term on s_i is not linear. Thus, this problem is essentially to minimize the mean squared error between the obtained registers toggling states $\mathbf{X}$ and the calculated power trace P_{real}, which is equivalent to compute the pseudo-inverse $\mathbf{X}^+$ so that $\boldsymbol{\beta} := \mathbf{X}^+ \cdot P_{real}$. The resulted $\boldsymbol{\beta}$ is applied on the input training data $\mathbf{X}$ and P_{real} for calibration, or applied on the new target input for prediction.

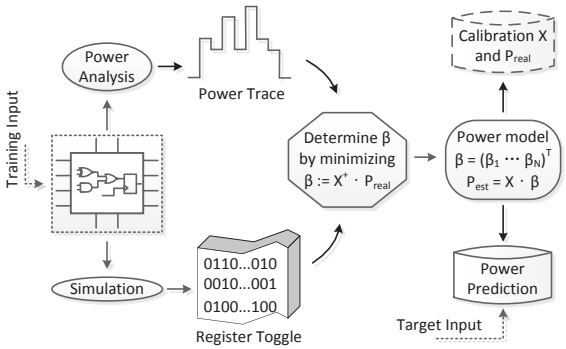

Figure 2: Power model abstraction methodology

3.2 Problem Formulation

Consider a circuit with n registers, simulated for m clock cycles to observe the registers toggling profile which can be denoted as a spare Boolean matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$:

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_i \\ \vdots \\ \mathbf{x}_m \end{pmatrix} = \begin{pmatrix} x_{11} & \cdots & x_{1j} & \cdots & x_{1n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{i1} & \cdots & x_{ij} & \cdots & x_{in} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mj} & \cdots & x_{mn} \end{pmatrix} \quad (4)$$

where each row $\mathbf{x}_i \in \mathbb{R}^{1\times n}$ represents the transition states between each two neighboring cycles of all registers, $i = 1, 2, ..., m$ and $j = 1, 2, ..., n$, that is, x_{ij} is 1 if the register j toggles between cycle $i-1$ and cycle i, otherwise x_{ij} is 0. Power consumption is calculated by gate level simulation for m clock cycles, denoted as a column vector $\mathbf{p} \in \mathbb{R}^{m\times 1}$ while $\mathbf{p} = \begin{bmatrix} p_1 & \cdots & p_i & \cdots & p_m \end{bmatrix}^T$ and p_i is the power value of i-th cycle for $i = 1, 2, ..., m$.

Since the power trace is strongly correlated to the registers toggling profile, a linear regression model is employed to describe the relationship between them. The coefficients for representing the contribution of the registers to the total power consumption are denoted as a column vector $\boldsymbol{\beta} \in \mathbb{R}^{n\times 1}$ while $\boldsymbol{\beta} = \begin{bmatrix} \beta_1 & \cdots & \beta_j & \cdots & \beta_n \end{bmatrix}^T$ and β_j is the contribution of j-th register to the total power consumption for $j = 1, 2, ..., n$. If the least square regression can be optimally performed, it will result in $\sum_{j=1}^{n} x_{ij}\beta_j = p_i$ of m linear equations and written in matrix form as

$$\mathbf{X}\boldsymbol{\beta} = \mathbf{p} \quad (5)$$

It is usually meaningful only when $m > n$, which is considered as an over determined system. Such a system usually has no exact solution, and consequently the goal is instead to find the coefficients $\boldsymbol{\beta}$ which fit the equations best, in the sense of solving the quadratic minimization problem, i.e., the linear least square minimization problem

$$F(\boldsymbol{\beta}) = \sum_{i=1}^{m} \left| p_i - \sum_{j=1}^{n} x_{ij}\beta_j \right|^2 = \|\mathbf{p} - \mathbf{X}\boldsymbol{\beta}\|^2 \quad (6)$$

where $F(\boldsymbol{\beta})$ is the objective function to be minimized.

In general, there are many linear dependent components and many redundant/random noise among the above re-

gression problem. A general approach to the least squares problem can be described by orthogonal projection theory [16]. Performing singular value decomposition (SVD) $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, where $\mathbf{U} \in \mathbb{R}^{m\times m}$ and $\mathbf{V}^T \in \mathbb{R}^{n\times n}$ are orthogonal matrices, and $\mathbf{\Sigma} \in \mathbb{R}^{m\times n}$ is a diagonal matrix (singular value). The pseudo-inverse of $\mathbf{\Sigma}$ is easily obtained by inverting its non-zero diagonal entries. Hence

$$\mathbf{X}\mathbf{X}^+ = \mathbf{X}\mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^T = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T \quad (7)$$

where $\mathbf{\Lambda}$ is obtained from $\mathbf{\Sigma}$ by replacing its non-zero diagonal entries with ones. Since $\mathbf{X}$ and $\mathbf{\Sigma}$ are obviously of the same rank, then $\mathbf{X}\mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^T = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$ is an orthogonal projection onto the image (column-space) of $\mathbf{X}$, and

$$\boldsymbol{\beta} = \mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^T\mathbf{p} \quad (8)$$

is a solution of the above least squares problem.

3.3 Model Abstraction with SVD

The solution with SVD method is the most computationally intensive, but indeed SVD is useful to get rid of redundant data, that is, for dimensionality reduction. The most important property of SVD is that we can take the first k largest singular values and construct an approximated rank-k matrix of $\mathbf{X}$ as following [17]

$$\mathbf{X}_k = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^T = \hat{\mathbf{X}} \quad (9)$$

where $\mathbf{U}_k \in \mathbb{R}^{m\times k}$, $\mathbf{\Sigma}_k \in \mathbb{R}^{k\times k}$ and $\mathbf{V}_k^T \in \mathbb{R}^{k\times n}$ are reduced matrices of $\mathbf{U}$, $\mathbf{\Sigma}$ and $\mathbf{V^T}$ respectively. Only the k column vectors of $\mathbf{U}$ and k row vectors of $\mathbf{V^T}$ corresponding to the k largest singular values $\mathbf{\Sigma}_k$ are calculated, and consequently much efficient than full SVD when $k \ll r$, where r is the rank of $\mathbf{X}$. The truncated SVD is no longer an exact decomposition of $\mathbf{X}$, but leads to an good approximation $\hat{\mathbf{X}}$. Actually, $\mathbf{X}_k$ is the best rank-k approximation of $\mathbf{X}$ in case of squared error loss. In other words, $\mathbf{X}_k$ is the one that minimizes the Frobenius norm

$$\|\mathbf{X} - \mathbf{X}_k\| = \sum\nolimits_i \sum\nolimits_j \left[\mathbf{X}\left(i,j\right) - \mathbf{X}_k\left(i,j\right)\right]^2 \quad (10)$$

The truncated SVD is essentially an approach for principal component analysis (PCA) to convert a set of observations of possibly corrected variables into a set of values of linearly uncorrected variables which is called principal components. Moreover, truncated SVD enable us to remove noises and linear dependent elements by using the most significant singular values. Thus, the coefficients $\boldsymbol{\beta}$ is approximately obtained as $\hat{\boldsymbol{\beta}}$:

$$\hat{\boldsymbol{\beta}} = \mathbf{V}_k\mathbf{\Sigma}_k^+\mathbf{U}_k^T\mathbf{p} \quad (11)$$

according to the the training cycles, and subsequently adopted to predict the power consumption in the future cycles.

3.4 RTL Instrumentation

The power estimation methodology includes calibration phase and prediction phase. Power calibration is to validate the obtained power model on the training clock cycles to check whether it is accurate enough. Power prediction is to apply the obtained power model for prediction on future clock cycles. The main framework contains 4 stages:

(i) Training cycles: perform simulation to dump registers toggling profile $\mathbf{X}$ and report power trace $\mathbf{p}$;
(ii) Build power model: perform truncated SVD $\mathbf{X}_k = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^T = \hat{\mathbf{X}}$ to compute coefficients $\hat{\boldsymbol{\beta}} = \mathbf{V}_k\mathbf{\Sigma}_k^+\mathbf{U}_k^T\mathbf{p}$;
(iii) Power trace calibration on training cycles $\langle \mathbf{p}, \mathbf{X}\cdot\hat{\beta}\rangle$;
(iv) Power trace prediction on future cycles: RTL simulation to obtain $\mathbf{X}'$, then compute $\mathbf{p}' = \mathbf{X}'\cdot\hat{\beta}$.

The proposed approach has significant potential for power estimation acceleration into commercial tools while the hardware-assisted verification is not novel for EDA tools [18]. Meanwhile, the abstracted power model is implemented as RTL instrumentation to report the real-time power trace by observing the selected registers toggling profile. This kind of power emulation is significantly critical for dynamically monitoring and on-line power management. Notice that the abstracted coefficients from equation (11) is usually floating-point number both in positive and negative, we consider to represent them as binary numbers by quantization and then map them into hardware prototyping platform. The quantization procedure is listed as follows:

(i) Scaling the coefficients with a proper threshold;
(ii) Round them towards nearest integer;
(iii) Convert decimal integer to binary string;
(iv) Instrument into RTL and synthesis on the hardware prototyping;
(v) Scaling down the estimated total power.

As demonstrated in Section 4.1, the required number of bits for representing the coefficients actually is not large because only very small amount of coefficients are relative large while most of them just require only two or three bits or even zero bit.

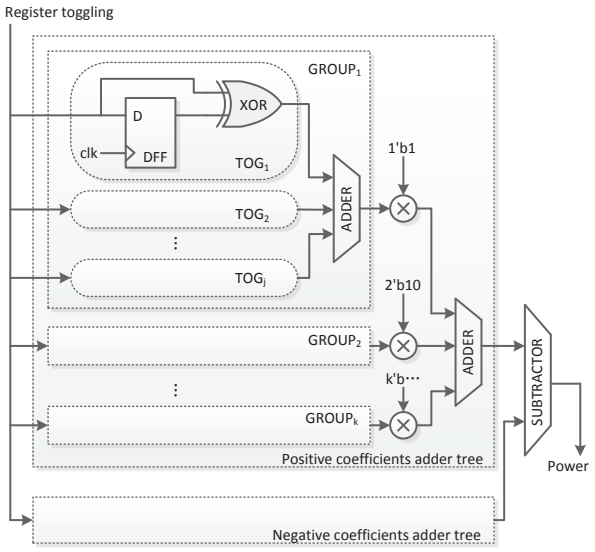

Figure 3: Block diagram of model implementation

The detailed block diagram of power model implementation is shown in Figure 3. Each term in the selected registers takes only one control signal as an input. For the previous clock cycle, the register state has been stored and locked. For the end of the next clock cycle, each toggling state is calculated by performing XOR operation between previous locked register state and current register state. The obtained toggling states are adopted as control signals to decide whether the corresponding coefficients should be added or not. For clarity, the selected registers are classified into several groups according to the required bits number after coefficients quantization. The registers from same group have the same coefficient. Thus, the toggling states are first accumulated in each group and then multiplied with their common coefficients. Naturally, the total power consumption is the sum of the multiplication results from all groups. Synthesized results in Section 4.1 will demonstrate that very small amount of additional logic resources are required to map the abstracted power model, because only

one XOR logic is required for each selected register and the operations for adding all terms can be optimized according to shared coefficients for each group.

4. EXPERIMENTAL RESULTS

4.1 Experimental Setup

The proposed methodology is implemented with Matlab scripts and evaluated on three open source design: H.264/AVC baseline decoder of QCIF resolution [19], AES IP Core [20] and AC 97 Controller IP Core [21]. The detailed tools flow of power model abstraction is demonstrated in Figure 4. The design cores are synthesized using Synopsys Design Compiler. Register toggling profile is dumped using full RTL level simulation. The gate level power is calculated using Synopsys PrimeTime PX with VCD files generated from full gate level simulation. For the sake of clarity, H.264/AVC is first chosen as an example for demonstrating the evaluation details in this subsection. Three video clips [22] of QCIF format are encoded using JM94 encoder [23] as input. For each case, 400K clock cycles are evaluated while the first 200K cycles are adopted as training data to run once for generating power models and he remaining 200K cycles are used for power prediction.

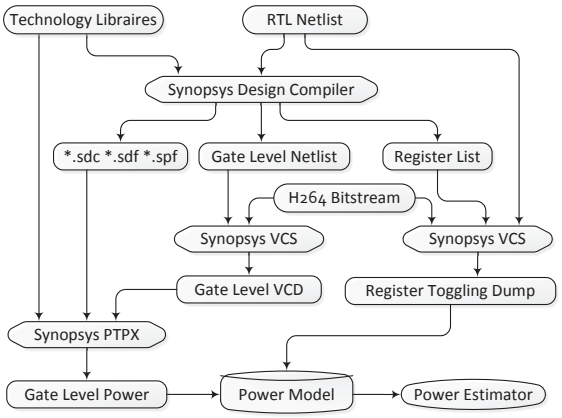

Figure 4: Tools flow of power model abstraction

Usually there are numerous registers within a design, but only part of the most active registers should have significant contribution to total power dissipation. The proposed model abstraction can automatically figure them out without any manual or incremental consideration. After obtaining the accurate power trace from PTPX, truncated SVD technique is employed to build the power model (100 largest singular values considered). The H.264/AVC encoder design has totally 31295 registers. The coefficients quantization results from 3 training cases are listed in Table 1. Respectively, there are 2050, 1655 and 2018 registers are kept for power estimation. From the Table 1, only several coefficients require 7 bit, 8 bit or 9 bit, and most of the coefficients just require 2 bit, 1 bit or even 0 bit.

Aiming to validate the promising performance of the proposed approach, the H.264/AVC encoder and corresponding power model in Figure 3 are implemented on FPGA prototyping platform. All of them are synthesized with Synopsys Synplify Pro for the target of Xilinx Virtex6 XC6VLX75T. Take Akiyo case as training data, the encoder takes up 22807 LUTs (48%) and power model takes up 3603 LUTs (7%) if 1-stage adder tree is employed in Figure 3. For the evaluation performance, the decoder is allowed to run at 41.6MHz while our power model can run at 73.3MHz which is enough for run-time estimation. If 2-stage adder tree is further utilized, about 5735 LUTs (12%) will be required and the run frequency is improved to 114.2MHz.

Table 1: Required bit number for coefficients quantization. Akiyo, Carphone and Claire totally requires 2050, 1655 and 2018 respectively.

Bit Num.	1	2	3	4	5	6	7	8	9
Akiyo	1242	503	150	70	50	21	11	2	1
Carphone	982	414	112	68	39	26	11	2	1
Claire	1108	634	142	54	42	23	12	2	1

4.2 Measuring Waveform Similarity

There are many approaches to evaluate the estimation accuracy, but we are much more interested in the similarity between the predicted waveform and the accurate waveform. In our work, the filtering method is exploited as a new metric to measure the waveform similarity. Because we are much more interested in the realistic response of a system while it is usually decided by its characteristic impedance, such as a RC/RLC network. But the equivalent network is unknown to designers before physical implementation. Suppose that there is an equivalent filtering network attached to the system for measuring the similarity, such as low pass window filter for removing the high frequency noises. The estimation accuracy is measured based on the filtered results by cycle-by-cycle error and relative error. Relative error is the average power consumption error. Cycle-by-cycle error is measured by normalized root mean square error (NRMSE) [24]. That is to compute the solution error of each cycle between the predicted power trace $\hat{p}_i$ and accurate values p_i obtained from PTPX.

$$NRMSE = \frac{RMSE}{p_{\max} - p_{\min}} \cdot 100\% \qquad (12)$$

while $RMSE = \sqrt{\sum_{i=1}^{m} (p_i - \hat{p}_i)^2 / m}$ is to represent the root mean square error.

4.3 Power Trace Calibration and Prediction

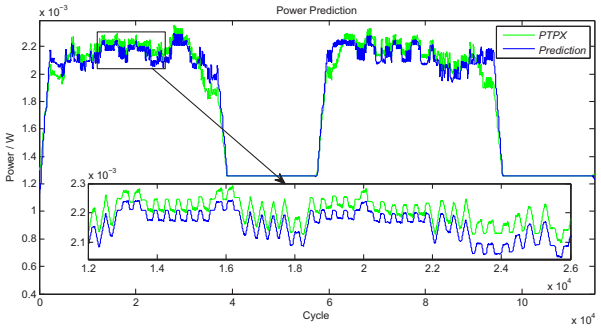

Figure 5: Power trace Comparison with window filter (window size = 2000). Greed curve: PTPX power. Blue curve: predicted power.

For each test video clip with several frames, the proposed power model is obtained from the training data with the first 200K cycles and calibrated for itself. For calibration on the three cases, their NRMSE is 2.36%, 2.34%, and 2.38% respectively, while the relative errors are 0.14%, 0.11% and 0.13% respectively. The power model is evaluated on the same video clip for future cycles, which is called self-prediction. Meanwhile, the power model is evaluated

on other video clips for future cycles, which is called cross-prediction. For the evaluated 3 video clips, their NRMSE is listed in Table 2 for cycle-by-cycle prediction. It has validated the efficiency and the accuracy of the proposed power model abstraction. From Table 2, there are less than 3% errors for self-prediction, and less than 5% errors for cross-prediction. Also the relative errors are list in Table 3, from which we can conclude that it is extremely accurate for both self-prediction and cross-prediction. Figure 5 shows cycle-by-cycle power waveforms while the power model obtained from Akiyo case is applied to predict the power consumption of Carphone case. The green curve is the power trace obtained from PTPX as accurate solution and the blue curve is the predicted power trace. For clarity, the results are plotted with a window filter while the window size is 2000.

Table 2: Normalized RMS error of cycle-by-cycle prediction

NRMSD	Akiyo	Carphone	Claire
Akiyo	2.51%	2.68%	3.20%
Carphone	4.35%	2.53%	4.13%
Claire	3.43%	3.62%	2.22%

Table 3: Relative errors of total power prediction

Relative Error	Akiyo	Carphone	Claire
Akiyo	0.09%	1.07%	1.89%
Carphone	2.58%	0.24%	3.47%
Claire	0.40%	1.19%	0.29%

In addition, the proposed model abstraction is evaluated on the AES case and AC97 case for power calibration and prediction. AES core includes 678 registers. AC97 core includes 2288 registers. Table 4 illustrates the relative error and normalized RMS error, from which we can conclude that the modeling accuracy is still guaranteed for additional design cases. And without loss of generality, the proposed model abstraction approach could be applicable to more general or complex designs for RTL instrumentation.

Table 4: Evaluation results for AES and AC97

IP Core	Calibration		Prediction	
	NRMSE	RelErr	NRMSE	RelErr
AES	3.25%	3.39%	3.35%	2.45%
AC97	1.74%	0.27%	0.85%	0.75%

5. CONCLUSIONS

In this paper, we propose a model abstraction approach based on the critical registers using singular value decomposition in the manner of machine learning. By adopting the circuit toggling activity and the power trace obtained from simulation as training data, it can abstract the principle components of the relationship between them. The abstracted power model can be easily implemented on hardware prototyping platform and provide real-time power estimation directly according to the critical registers toggling profile. The proposed critical registers power model with SVD technique is shown to be extremely efficient and accurate for power calibration and real-time prediction. In the future work, we will consider more complex circuit components as observation candidates and validate the proposed power model abstraction.

6. REFERENCES

[1] International Technology Roadmap for Semiconductors, 2012 Edition. http://www.itrs.net/Links/2012ITRS/Home2012.htm.

[2] Synopsys PrimeTime, Synopsys Inc. http://www.synopsys.com/Tools/Implementation/SignOff/Pages/PrimeTime.aspx.

[3] PowerArtist, Apache Design, Inc. http://www.apache-da.com/products/powerartist/powerartist-xp.

[4] PowerPro, Calypto Design Systems, Inc. http://calypto.com/en/products/powerpro/overview.

[5] SpyGlass Power, Atrenta Inc. http://www.atrenta.com/solutions/spyglass-power.htm5.

[6] David Brooks, Vivek Tiwari, and Margaret Martonosi. Wattch: a framework for architectural-level power analysis and optimizations. In *Proc. ISCA*, pages 83–94, 2000.

[7] Chen-Wei Hsu, Jia-Lu Liao, Shan-Chien Fang, Chia-Chien Weng, Wen-Tsan Hsieh Shi-Yu Huang, and Jen-Chieh Yeh. PowerDepot: integrating IP-based power modeling with ESL power analysis for multi-core SoC designs. In *Proc. DAC*, pages 47–52, 2011.

[8] Stefan Schürmans, Diandian Zhang, Dominik Auras, Rainer Leupers, Gerd Ascheid, Xiaotao Chen, and Lun Wang. Creation of ESL power models for communication architectures using automatic calibration. In *Proc. DAC*, Article No. 58, 2013.

[9] Dam Sunwoo, Gene Y. Wu, Nikhil A. Patil, and Derek Chiou. PrEsto: An FPGA-accelerated power estimation methodology for complex systems. In *Proc. FPL*, pages 310–317, 2010.

[10] Joel Coburn, Srivaths Ravi, and Anand Raghunathan. Power emulation: a new paradigm for power estimation. In *Proc. DAC*, pages 700–705, 2005.

[11] Abhishek Bhattacharjee, Gilberto Contreras, and Margaret Martonosi. Full-system chip multiprocessor power evaluations using FPGA-based emulation. In *Proc. ISLPED*, pages 335–340, 2008.

[12] Sumit Ahuja, Avinash Lakshminarayana, and Sandeep Kumar Shukla. *Low Power Design with High-Level Power Estimation and Power-Aware Synthesis*, chapter Regression-Based Dynamic Power Estimation for FPGAs. Springer, 2012.

[13] Alessandro Bogliolo, Luca Benini, and Giovanni De Micheli. Regression-based RTL power modeling. *ACM Transactions on Information Systems*, 10(4):320–344, October 1992.

[14] Sumit Ahuja, Deepak A. Mathaikutty, and Sandeep K. Shukla. Model-based power estimation using least squares regression on FSMD models. *Technical Report*, Virginia Tech., 2008.

[15] Michael Eiermann and Walter Stechele. Novel modeling techniques for RTL power estimation. In *Proc. ISLPED*, pages 323–328, 2002.

[16] Charles L. Lawson and Richard J. Hanson. *Solving Least Squares Problems*. SIAM in Applied Mathematics, Philadelphia, 1974.

[17] G. W. Stewart. On the early history of the singular value decomposition. *SIAM Review*, 35(4):551–566, December 1993.

[18] Hardware based simulation engine for SoC Verification, Synopsys Inc. http://www.synopsys.com/Tools/Verification/hardware-verification/Pages/default.aspx/.

[19] H.264/AVC Baseline Decoder. http://opencores.org/project,nova.

[20] AES IP Core. http://opencores.org/project,aes_core.

[21] AC 97 Controller IP Core. http://opencores.org/project,ac97.

[22] Xiph.org Video Test Media [derf's collection]. http://media.xiph.org/video/derf/.

[23] JM94 Encoder for H.264/AVC. http://iphome.hhi.de/suehring/tml/.

[24] J. Scott Armstrong and Fred Collopy. Error measures for generalizing about forecasting methods: Empirical comparisons. *International Journal of Forecasting*, 8(1):69–80, June 1992.