ICT-ENERGY LETTERS

Evaluation of hybrid run-time power models for the ARM big.LITTLE architecture

Krastin Nikov, Jose Nunez-Yanez
University of Bristol, United Kingdom

***Abstract*—** Heterogeneous processors, formed by binary compatible CPU cores with different microarchitectures, enable energy reductions by better matching processing capabilities and software application requirements. This new hardware platform requires novel techniques to manage power and energy to fully utilize its capabilities, particularly regarding the mapping of workloads to appropriate cores. In this paper we validate relevant published work related to power modelling for heterogeneous systems and propose a new approach for developing run-time power models that uses a hybrid set of physical predictors, performance events and CPU state information. We demonstrate the accuracy of this approach compared with the state-of-the-art and its applicability to energy aware scheduling. Our results are obtained on a commercially available platform built around the Samsung Exynos 5 Octa SoC, which features the ARM big.LITTLE heterogeneous architecture.

I.INTRODUCTION

IN this paper we have explored, tested and improved published power model generation methodologies on a big.LITTLE development platform. Our focus is real-time usability in a dynamic system so we focused on techniques to predict runtime power that do not have substantial overheads and still achieve respectable accuracy. We also investigate the tradeoff between model complexity/accuracy and have developed a scalable approach with 3 different groups of predictors.

II.RELATED WORK

Modern power models use high-level events to derive power, since they are much easier to observe. Takouna et al. [1] present a very minimal linear power model for the Intel Xeon E5540 CPU, which uses frequency and number of cores to predict power with an average of 7% error. A limitation to this approach is that it cannot predict the changes in power consumption at a particular frequency level, as it will only model the average power for that frequency.

A successful way to observe finer changes in program execution and power is to use hardware system information available from the Performance Monitoring Units on a CPU. Pricopi et al. [2] use PMU information to develop complex models for predicting performance on a big.LITTLE SoC, however, they have also built an accurate empirical model for the big.LITTLE ARM Cortex-A15 CPU using 7 specific PMU events. They report around 2.6% average error. Another approach to predicting power is using information about the CPU state and utilization.

Walker et al. [3] present a CPU frequency and utilization based model for a big.LITTLE platform, using information about CPU time spent in idle available from the device OS. Tested on the same workload as the PMU model, the CPU frequency and idle time model achieves 10.4% and 8.5% error for the ARM Cortex-A7 and ARM Cortex-A15 respectfully.

III.METHODOLOGY

III.1.Development platform and workload

We have developed and tested our methodology on the ODROID XU3 development platform. It has an ARM big.LITTLE chip, which has four ARM Cortex-A15 (the big) CPU cores and four ARM Cortex-A7 (the LITTLE) CPU cores. The board also has four sensors, measuring the A15, A7, RAM and GPU power, current and voltage. The platform was set up with a Lubuntu 12.04 running the latest kernel available from the board support team.

The ODROID XU3 has a broad DVFS range with the Cortex-A15 having 18 available frequency levels and the Cortex-A7 with 14 available frequency levels from 0.2-1.4 GHz. The ideal workloads for this system would be exhaustive benchmarks with diverse behavior in order to capture different scenarios and long runtime, which is why we used cBench that exhibit these characteristics.

We have observed a significant range in power consumption for both CPU core types while running the workload. We recorded an average power variation of up to 126% at 1.6 GHz and 143% at 0.8 GHz for the Cortex-A15 and Cortex-A7 respectively.

III.2.Model generation and verification

The workload microbenchmarks are randomly split in half into a train and test set. After we collect the results from the train set we use linear regression in the form of Ordinary Least Squares to compute the predictor equation:

$$\beta = (X^T X)^{-1} X^T y = \left(\sum x_i x_i^T\right)^{-1} \left(\sum x_i y_i\right)$$

Power is used as the dependent variable β in the above equation, also known as regressand. The events are expressed as the X vector of independent variables, a.k.a. regressors. The OLS method outputs a vector β, which holds coefficients extracted from the activity vectors. Then the equation:

$$P_{CPU} = \beta_0 + (\beta_1 \cdot event_1) + (\beta_2 \cdot event_2) + \cdots (\beta_n \cdot event_n)$$

is used to estimate power usage using a new set of events. We evaluate the accuracy of the modelled equation by using the collected events from the test set and measure the percent difference (error) between the measured power and the estimated power by plugging the new events in the equation.

The advanced model we have developed consists of 3 distinct components:

1. Physical, which has physically controlled regressands such as CPU voltage, CPU frequency and CPU temperature, expressed in Equation 1 shown below.

$$P(W) = const. + (\alpha_1 \cdot CPU\ Voltage) + (\alpha_2 \cdot CPU\ Frequency) + (\alpha_3 \cdot CPU\ Temperature) \quad (1)$$

2. PMU events, expressed in Equation 2. We use consider several events common to both CPUs and after mathematical analysis choose the top 5 most correlated to power.

$$P(W) = const. + (\alpha_1 \cdot Cycles) + (\alpha_2 \cdot L1\ D\ Access) + (\alpha_3 \cdot L1\ I\ Access) + (\alpha_4 \cdot Instructions) + (\alpha_5 \cdot Mem\ Access) \quad (2)$$

3. CPU state, expressed in Equation 3, uses information about time spent in different relevant CPU states, excited during workload execution.

$$P(W) = const. + (\alpha_1 \cdot CPU\ User) + (\alpha_2 \cdot CPU\ System) + (\alpha_3 \cdot CPU\ Idle) + (\alpha_4 \cdot CPU\ IO\ Wait) + (\alpha_5 \cdot CPU\ IRQ) + (\alpha_6 \cdot CPU\ Soft\ IRQ) \quad (3)$$

We then combine the events from the 3 component models listed above to develop a Physical + PMU events + CPU state (called P2S) model. The developed composite model P2S is compared to relevant published work, listed in Section II. The model presented by Takouna et al. [1] from the University of Potsdam is listed as *UoP* model for short. Walker et al. [3] from University of Southampton are listed as *UoS full* model in our results and we call Pricopi et al. [2] from Cambridge Silicon Radio *CSR* model. We have also compared against an *Updated CSR* model for the Cortex-A15, extended with physical and CPU state information.

IV.RESULTS

IV.1.Full frequency range models

We first obtain results using the models from our own methodology and evaluate the performance of the individual component models as well as the P2S model. Our observations show that the resulting complex P2S model is not always the best performer, instead the best full frequency model for the Cortex-A15 and the Cortex-A7 is Physical with an average accuracy of 19.95% and 10.46% respectively over the respective frequency ranges.

IV.2.Per-frequency models

Based on previous observations on the variability of PMU events and CPU state information between frequency levels we decided to try and calculate the models on a per-frequency basis. We see a large reduction for the average model error and that adding CPU state and PMU event predictors improve accuracy significantly compared to just using the Physical model. Overall we report and average of 8% error for the ARM Cortex-A15 and 5% error for the per frequency level P2S power model, which is on our target goal for the Cortex-A7.

IV.3.Inter-core Models

We have also explored the usability of the developed P2S model to accurately predict inter-core power. This is done in order to see if the modelling methodology can be used to aid scheduling decisions for migrating a task from one core type to the other. The cross prediction model for the Cortex-A15 CPU power, trained and tested using Cortex-A7 events, comes really close to the P2S per-frequency model, both achieving 8% average prediction error. The trained cross-prediction model for the Cortex-A7 achieves a respectable accuracy of 9.26%.

IV.4.Experimental Evaluation

This section describes our results after testing some the published models described previously using our workflow on the ODROID XU3. Figure 1 and 2 show them compared to our best performing per-frequency model P2S for the Cortex-A15 and Cortex-A7 respectively. In both cases the P2S model performs a lot better with the exception of the Updated CSR model. This is to be expected since the Updated CSR is a more complex model with larger, more specialized list of PMU events that are used in the model.

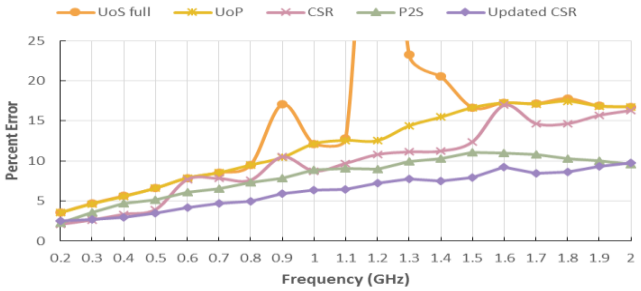

Fig. 1. Comparison between P2S and other published models on the Cortex-A15

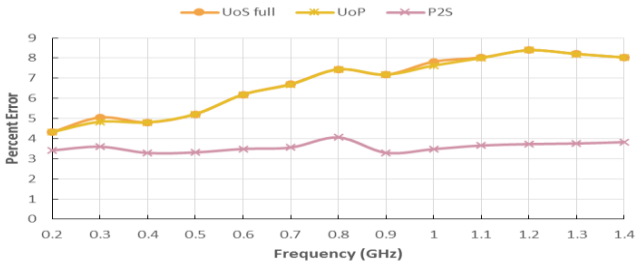

Fig. 2. Comparison between P2S and other published models on the Cortex-A7

V.CONCLUSIONS

This paper has presented a novel hybrid approach for run-time power prediction and modelling in heterogeneous single ISA big.LITTLE processors. The developed models are compared to other published work in the area using the same methodology and show improved accuracy. We have tried to predict the power consumption for all possible voltage/frequency levels for the two available CPU types and conclude that using per-frequency level models significantly improve accuracy compared to a unified frequency model. Our results show respectable accuracy with an average 8% prediction error on the ARM Cortex-A15 and 5% prediction error on the ARM Cortex-A7. We also show that our methodology can also be used for accurate cross-prediction models for both CPU cores.

All our work is open source and can be viewed and downloaded at https://github.com/kranik/ARMPM

ACKNOWLEDGEMENTS

This work is supported by ARM research funding and the University of Bristol.REFERENCES

[1] I. Takouna, W. Dawoud, and C. Meinel, “Accurate Mutlicore Processor Power Models for Power-Aware Resource Management,” *2011 IEEE Ninth Int. Conf. Dependable, Auton. Secur. Comput.*, pp. 419–426, Dec. 2011.

[2] M. Pricopi, T. S. Muthukaruppan, V. Venkataramani, T. Mitra, and S. Vishin, “Power-performance modeling on asymmetric multi-cores,” *2013 Int. Conf. Compil. Archit. Synth. Embed. Syst.*, pp. 1–10, Sep. 2013.

[3] M. J. Walker, A. K. Das, G. V Merrett, and B. M. Al-hashimi, “Run-time Power Estimation for Mobile and Embedded Asymmetric Multi-Core CPUs.”

[4] I. Takouna, W. Dawoud, and C. Meinel, “Accurate Mutlicore Processor Power Models for Power-Aware Resource Management,” *IEEE Ninth Int. Conf. Dependable, Auton. Secur. Comput.*, pp. 419–426, Dec. 2011.

[6] M. Pricopi, T. S. Muthukaruppan, V. Venkataramani, T. Mitra, ç and S. Vishin, “Power-performance modeling on asymmetric multi-cores,” *2013 Int. Conf. Compil. Archit. Synth. Embed. Syst.*, pp. 1–10, Sep. 2013.

[7] M. J. Walker, A. K. Das, G. V Merrett, and B. M. Al-hashimi, “Run-time Power Estimation for Mobile and Embedded Asymmetric Multi-Core CPUs.”