# A Run-Time Power Analysis Method using OS-Observable Parameters for Mobile Terminals

Kaneda, Yusuke
Department of Electrical Engineering and Computer Science, School of Engineering, Kyushu University

Okuhira, Takumi
Department of Electrical Engineering and Computer Science, School of Engineering, Kyushu University

Ishihara, Tohru
System LSI Research Center, Kyushu University

Hisazumi, Kenji
System LSI Research Center, Kyushu University

他



出版情報：International Conference on Embedded Systems and Intelligent Technology. 2010 (1), pp.39-, 2010-02
バージョン：
権利関係：

# A Run-Time Power Analysis Method using OS-Observable Parameters for Mobile Terminals

Yusuke KANEDA*, Takumi OKUHIRA*, Tohru ISHIHARA†, Kenji HISAZUMI†,
Takeshi KAMIYAMA‡ and Masaji KATAGIRI‡
* Dept. of EE and CS, School of Eng., Kyushu Univ. 744 Motooka, Nishi-ku, Fukuoka, 819-0395 Japan
†System LSI Research Center, Kyushu Univ. 3-8-33 Momochihama, Sawara-ku, Fukuoka, 814-0001 Japan
‡Research Laboratories, NTT DOCOMO, Inc., 3-5, Hikarinooka, Yokosuka, Kanagawa, 239-8536 Japan
E-mail: *{kaneda,okuhira}@soc.ait.kyushu-u.ac.jp, †{ishihara,nel}@slrc.kyushu-u.ac.jp,
‡{kamiyamat,katagirim}@nttdocomo.co.jp

***Abstract*—This paper presents a lightweight power consumption model and its generation method for quickly and accurately analyzing the power consumption of wireless communication devices. Many power analysis methods for VLSI circuits have been proposed before. However, most of them are based on hardware simulation which is very slow and power consuming. Therefore, they cannot be applied for analyzing the run-time power consumption of battery powered devices where the low power consumption is the most important criterion. In our method, the power analysis can be done with a small overhead. Experimental results with an N810 terminal developed by Nokia, and an Android Dev Phone by HTC demonstrate that the error of our power analysis method is on an average 4.33% compared to the measured power results. Once the power consumption model has been developed for a target device, the power consumption of application programs running on the device can be analyzed in real-time with a small power overhead.**

## I. Introduction

The increasing demand for wireless communication devices such as cellular phones and PDAs makes low power consumption one of the most important criteria in embedded computer systems. In many years ago, the power consumption of computer systems is mostly depending on hardware. Therefore, many researches for saving the power consumption of the computer systems are focused on modifying the hardware design so that the hardware dissipates less power. However, in today's wireless communication devices, the power consumption strongly depends on software running on the devices since most functions are implemented in the software. As more and more functions are implemented in software, it is becoming more important to optimize software for reducing the power consumption. The optimization can be done by modifying the algorithm implemented in the software or by using compilation techniques for optimizing the object code. Another important factor which strongly affects the power consumption of today's wireless communication devices is end-user's usage. Since today's smart phones employ many sophisticated functions which have similar functionalities with different power consumptions from each other, the power consumption of devices strongly depends on how those functions are used by end-users. Hence, it is becoming more important to optimize the user's usage for reducing the energy consumption. The energy saving can be done by selecting appropriate functions and options of devices so that the target device consumes less power. To make this possible, analyzing the run-time power consumptions corresponding to the functions and the options is very important.

This paper proposes a lightweight power consumption model which makes it possible to analyze the power consumption of wireless communication devices on the fly using a small number of dynamic parameters obtained through an operating system (OS). The dynamic parameters include CPU utilization, activity of wireless LAN, throughput of wireless data communication, and throughput of data read/write for a storage. A lot of techniques for analyzing the power consumption of computer systems have been proposed before. However, most of them are targeting CPU and main memory only. Our model targets entire device including CPU, memory, flash disks, RF circuits and other peripheral circuits. Many other previous techniques use parameters which cannot be obtained at run-time although the accuracy may be very good if those parameter values can be obtained. For example, it may be possible to accurately analyze the energy consumption of computer systems based on the number of instructions executed on a CPU core, the number of internal bus transitions, the number of cache memory accesses and so on. However, most of these parameter values cannot be obtained at run-time or is very time consuming to obtain. Our technique uses dynamic parameters, all of which are quickly obtained with a small overhead through an OS running on a target device. To the best of our knowledge, this is the first proposal which makes it possible to accurately analyze the power consumption of entire wireless communication devices at run-time using a small number of dynamic parameters quickly obtained through an OS. This paper also presents an automated flow for generating the power consumption model targeting for a given wireless communication device. Our method uses a small number of training benches which are suited for characterizing an individual impact of each parameter on the power consumption of a target device.

The rest of the paper is organized as follows. Section 2 summarizes related work. Our approach for generating the power consumption models for target wireless devices is presented in Section 3. Section 4 presents experiments and

results for two commercial smart phones. The paper concludes in Section 5.

## II. RELATED WORK

### A. Power Estimation Techniques

The most accurate and fastest approach to identify the energy consumption of a target device is to directly measure the average power consumption of the actual device. An embedded power meter like 4-T decay sensor [1] measures the power consumption of target devices on the fly. The major drawback of using on-chip power meter is that it involves a considerable overhead to collect the power consumption values from different subsystems (e.g., a memory system) and to send them to the CPU for analyzing the energy consumption of entire system in real-time. It is also very difficult to identify the power consumption of an individual software sub-process separately. To overcome these issues, several techniques exploiting a hardware performance counter have been proposed [2]–[6]. The performance counters separately count the number of hardware events like the number of CPU instructions executed, the number of memory accesses, and the number of bus transitions. These values are used as proxies for power meters. The main problem is that the performance counter only targets the hardware events related to a CPU. It does not count the number of events on peripheral circuits like RF circuits and flash disks. Unlike above techniques, our approach models the power consumptions of CPU, memory, flash disks and peripheral circuits like RF circuits with very small overhead in real-time. Bircher et al. proposes a power estimation method targeting for a complete system which includes CPU, memory, I/O, chipset and disk [7]. However, the problem is that many embedded processors do not employ the hardware counter which involves considerable power consumption to access. Our technique uses model-parameters whose run-time values can be quickly obtained through an OS with small overhead. Our technique also can accurately identify the power consumptions of individual software sub-processes separately. This helps end-users to analyze run-time power consumption process-by-process. In [8], a methodology to generate power model of processor systems including CPU core, accelerators, interfaces and peripherals is presented. They use OS-level parameters as model-parameters. However, they are not targeting a wireless terminal where the power consumption for the wireless communication is dominant. Also, the approach for generating the power model is quite different from ours.

In the past, a lot of simulation-based power estimation techniques have been proposed [9]–[15]. The idea is to exploit power-related parameter values obtained through the simulation to abstract the power consumption of a target system. The parameters include the number of CPU instructions executed, the number of cache hits/misses, the number of pipeline stalls and the number of branches taken. The accuracy of these methods can be improved by increasing the number of model-parameters accounting for the power consumption. The main drawback of these techniques is that they rely on time-consuming offline simulation for estimating the power consumption of a target system. These methods are useful if they are used in a design phase of hardware and software. However, these simulations are too slow and power-consuming for run-time power analysis. Unlike these methods, our method can estimate the run-time power consumption of the target system with a small number of model-parameters whose values can be quickly obtained through an OS.

PowerTop [16] developed by Intel identifies power hungry sub-processes running on a CPU. This makes it possible to visualize the run-time power consumption of an individual sub-process running on the CPU. However, this method targets microprocessors only. Our method extends this approach for estimating not only CPU power but also power consumptions related to disk reads/writes and wireless communications.

### B. Power Modeling Techniques

Most of existing power estimation techniques including the techniques presented in [2]–[8] [10]–[16] assume a linear approximation model for estimating the power consumption of a target system. However, none of them describes how to find model parameters and training benches which should be used for accurately characterizing the power consumption of a target system. In [17], Tan et al. modeled the software energy consumption using a linear equation and discussed the parameters required to accurately estimate the energy consumption. However, they did not provide any method to find training benches required for the accurate power characterization. In [9], Lee et al. present the importance of training benches for the modeling and propose a method for generating training benches suitable for the CPU power characterization. However, their approaches are targeting a CPU and memory sub-systems only. The fundamental issue of these previous approaches is that their model is limited to the linear equation which way lose an accuracy of the power estimation. Unlike those previous techniques, our approach characterizes the power consumption of mobile terminal using polynomial model. It is more general and can cover a variety of devices like DVFS-capable CPUs and RF circuits whose power consumptions may not be linear to their utilizations intuitively. Our approach characterizes the power consumption of entire mobile terminal including the power consumptions related to CPU computations, memory accesses, disk reads/writes and wireless communications.

## III. POWER CHARACTERIZATION

### A. Power Consumption Model

Generally the following three items, 1) model parameter, 2) model equation, and 3) training benches used for regression analysis, strongly affect the model accuracy. Therefore, these items should be determined very carefully. Since our goal is to accurately analyze the power consumption of entire device at run-time with a small overhead, it is also very important to obtain the values of model parameters quickly with a small power overhead through software running on a CPU. In our approach, the following six parameters are selected as

model parameters whose values are quickly obtained though an operating system running on a CPU core.

(A) CPU utilization
(B) throughput of wireless data reception (MB/s)
(C) throughput of wireless data transmission (MB/s)
(D) activity of wireless LAN
(E) throughput of data read from flash disk (MB/s)
(F) throughput of data write to flash disk (MB/s)

The CPU utilization represents the CPU time used per second. Similarly, the activity of wireless LAN represents the time period during which the wireless LAN is active per second. All of above parameters intuitively have large impacts on the power consumption. In this paper, parameters other than the above ones are regarded as a constant since they are out of the modeling target. In our current method, the power consumption of software sub-processes running on a CPU are estimated all together. However, it is technically possible to estimate the power consumption of individual sub-process separately since the parameters corresponding to the CPU utilizations of the sub-processes can be obtained separately.

As presented in the previous section, most of existing power estimation techniques assumes a linear model as a model equation since it is very simple and robust for an over-fitting problem. The over-fitting is generally observed in a model which is too complicated [18]. If we use polynomial or exponential equation, coefficients obtained by regression analysis with a small number of training patterns can be far from actual ones. Although the over-fitting problem is hard to occur for the linear model, it can be inaccurate if the power consumption of a specific device is not linearly related to any of model parameters. Therefore, in our modeling method, a polynomial model as shown in (1) is used. As a remedy of the over-fitting problem, we use an enough number of training benches for the modeling so that the over-fitting problem can be avoided. A least-square method is used for finding coefficients of the corresponding parameters. Hence, the power model is given by the following expression.

$$P_{est} = c_0 + c_1 \cdot P_{cpu}^{c_2} + c_3 \cdot P_{recep}^{c_4} + c_5 \cdot P_{trans}^{c_6} + c_7 \cdot P_{wact}^{c_8} + c_9 \cdot P_{read}^{c_{10}} + c_{11} \cdot P_{write}^{c_{12}} \quad (1)$$

where $P_{est}$ is the power consumption estimated. $P_{cpu}$, $P_{recep}$, $P_{trans}$, $P_{wact}$, $P_{read}$ and $P_{write}$ represent CPU utilization, throughput of wireless data reception, throughput of wireless data transmission, activity of wireless LAN, throughput of data read from flash disk, and throughput of data write into flash disk, respectively. $c_0$ to $c_{12}$ are coefficients to be determined through multiple regression analysis.

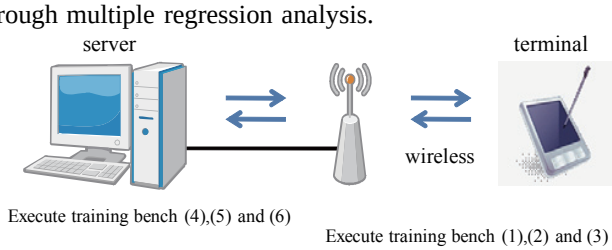



Fig. 1. Experimental Environment

### B. Training Bench and Environment for Characterization

As mentioned in the previous subsection, training benches used for regression analysis are very important for accurate model fitting. To get accurate results for multiple regression analysis, we make 6 training benches which respectively measure the influence of each parameter on the power consumption of the target device.

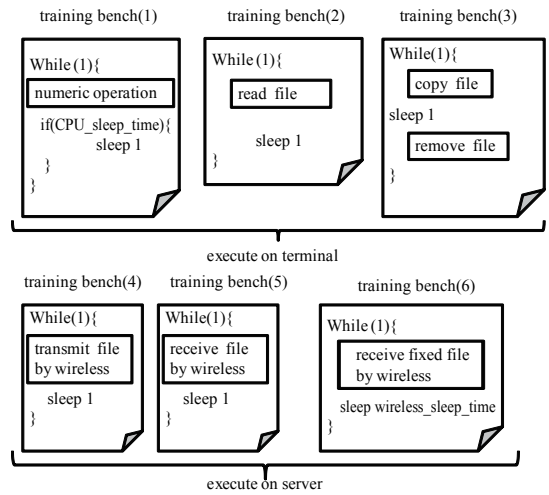



Fig. 2. Training Bench for multiple regression analysis

Fig.2 shows an example of training benches used for the characterization. Training benches (1), (2) and (3) as shown in Fig.1 are executed on the target mobile terminal. Training benches (4), (5) and (6) are executed on the server. By changing a control parameter of the training bench (1), CPU utilization of mobile terminal can be changed. Since most functions like wireless communications and disk accesses use the CPU power, there can be a strong correlation between the first model parameter (i.e., CPU utilization) and the other model parameters. If two or more model-parameters in a multiple regression analysis are highly correlated to each other, the coefficients obtained may change erratically in response to small changes in the model or the data. This statistical phenomenon is called as multicollinearity [18]. To obtain accurate model-coefficients through the multiple regression analysis, we tune control-parameters in the training benches so that inter-correlations among model-parameters can be lowered. For example, the training bench (1) is executed simultaneously with the other training benches to reduce the correlation between CPU utilization and the other parameters in a way that the CPU utilization is adjusted intentionally by changing sleep time of CPU (i.e., CPU_sleep_time). Training bench (2) aims to measure the independent impact of the throughput of disk-read on the power consumption. In the same way, training bench (3) measures the power consumption related to the throughput of a disk-write operation. Training benches (4) and (5) are used to characterize the power consumption related to the throughput of wireless data transferred between the server and the terminal by changing a size of the data transferred at a time. Training bench (6) enables to characterize

the power consumption related to the activity of wireless LAN by changing sleep time of the wireless LAN circuit (i.e., wireless_sleep_time). To eliminate interferences from other processes running on the terminal, training benches (4), (5) and (6) are executed on the server.

### C. Characterization Flow

An overview of a power characterization flow is shown in Fig.3. The characterization is done by multiple regression analysis using equation presented in Section III.A and above mentioned training benches. The power consumption values and parameter values are obtained using the following use-cases.

(I) Execute training bench (1) on the target terminal.
(II) Execute training benches (1) and (2) simultaneously on the target terminal.
(III) Execute training benches (1) and (3) simultaneously on the target terminal.
(IV) Execute training bench (4) on the server and training bench (1) on the terminal synchronously.
(V) Execute training bench (5) on the server and training bench (1) on the terminal synchronously.
(VI) Execute training bench (6) on the server and training bench (1) on the terminal synchronously.

The values of the total power consumption of the target terminal are obtained every 20 millisecond with a digital ammeter. Parameter values are obtained every second by checking system parameters like /proc/stat and /proc/net/dev through Linux OS. After obtaining power consumption values and parameter values, coefficients of the model are found using multiple regression analysis.

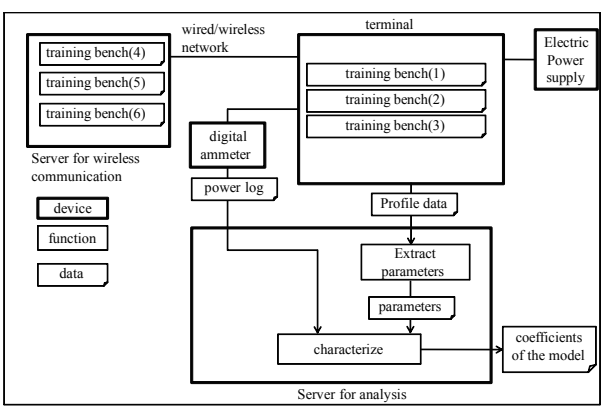



Fig. 3. Characterization Flow

## IV. CASE STUDY

### A. Target Devices

Nokia N810 and Android Dev Phone mobile terminals are used for our case study. The N810 is equipped with wireless LAN and the Linux-based Internet Tablet OS 2008. It employs OMAP2420 as a CPU core, 256MB RAM, 2GB NAND flash memory, and 8GB NAND flash memory. The Dev Phone employs a wireless LAN device and the Android OS. It has a MSM7201A CPU developed by Qualcomm Inc, 256MB ROM, 192MB RAM and 1GB NAND flash memory. Linux OS is installed on a 1GB flash memory. Note that we only use the wireless LAN of the terminal for the Internet access from the terminal.

### B. Experimental Setup

As mentioned in Section III, we use the CPU utilization, throughput of wireless data reception, throughput of wireless data transmission, activity of wireless LAN, throughput of data read from flash disk, and throughput of data write into flash disk as our model parameters. The values of those parameters are obtained from the /proc file system. Coefficients of corresponding parameters are found by multiple regression analysis based on the parameter values and the power consumption values measured with an ammeter. A least square method is used in the regression analysis.

Once the model is obtained, the power consumption can be estimated for any operating state of the terminal by obtaining the six parameter values mentioned above from the /proc file system. We evaluate the accuracy of the model by comparing these estimated values with actual measured power consumption values. We use seven benchmark programs selected from the MiBench suite and thirteen use-cases based on Wget as applications for the evaluation. Wget is a program which lets one download contents from the Internet in a number of effective ways. Three different types of data, 1) GNU/freefont/, 2) GNU/gcc/, and 3) gnu-0.2.tar.gz are used for the evaluation of Wget. The first one is a directory which contains many small files. The second one is also a directory containing middle size files. The last one is a large size file. For the experiment of Wget, the power consumptions dissipated during these three types of data are downloaded from different web sites are evaluated. We execute training benches and application programs on Linux OS. Note that the training benches and the application programs under evaluation are quite different from each other.

### C. Characterization Results

TABLE I shows values of coefficients of corresponding model-parameters obtained by the multiple regression analysis. $c_0$ to $c_{12}$ shown in the leftmost column of TABLE I respectively represent the coefficients described in Section

TABLE I
COEFFICIENT OF CORRESPONDING PARAMETERS

| Coefficient of parameters | N810 | Dev Phone |
|---|---|---|
| $c_0$ | 0.4650 | 0.5618 |
| $c_1$ | 0.5910 | 0.1422 |
| $c_2$ | 1.0472 | 1.7678 |
| $c_3$ | 0.4103 | 0.4829 |
| $c_4$ | 1.0311 | 1.0416 |
| $c_5$ | 0.4203 | 0.8101 |
| $c_6$ | 1.0256 | 1.2247 |
| $c_7$ | 0.2544 | 0.5490 |
| $c_8$ | 0.9928 | 1.2367 |
| $c_9$ | 0.0118 | 0.0247 |
| $c_{10}$ | 0.9015 | 0.7608 |
| $c_{11}$ | 0.0304 | 0.1285 |
| $c_{12}$ | 0.6822 | 0.3360 |

TABLE II
CORRELATION BETWEEN PARAMETER-1 AND -2

| Parameter-1 | Parameter-2 | correlation coefficient | |
|---|---|---|---|
| | | N810 | Dev Phone |
| $P_{cpu}$ | $P_{recep}$ | 0.091 | 0.113 |
| $P_{cpu}$ | $P_{trans}$ | 0.092 | 0.117 |
| $P_{cpu}$ | $P_{wact}$ | 0.232 | 0.336 |
| $P_{cpu}$ | $P_{read}$ | 0.115 | 0.129 |
| $P_{cpu}$ | $P_{write}$ | 0.105 | 0.084 |
| $P_{recep}$ | $P_{trans}$ | -0.071 | -0.084 |
| $P_{recep}$ | $P_{wact}$ | 0.628 | 0.538 |
| $P_{recep}$ | $P_{read}$ | -0.115 | -0.138 |
| $P_{recep}$ | $P_{write}$ | -0.102 | -0.093 |
| $P_{trans}$ | $P_{wact}$ | 0.673 | 0.549 |
| $P_{trans}$ | $P_{read}$ | -0.106 | -0.081 |
| $P_{trans}$ | $P_{write}$ | -0.106 | -0.094 |
| $P_{wact}$ | $P_{read}$ | -0.249 | -0.354 |
| $P_{wact}$ | $P_{write}$ | -0.230 | -0.265 |
| $P_{read}$ | $P_{write}$ | -0.091 | -0.031 |

III.A. Each coefficient shows an individual impact of the corresponding parameter on the power consumption of the target mobile terminal. TABLE II shows correlation coefficients. These values intuitively show risk levels of the multicollinearity problem. Generally, if a value of correlation coefficient between two model-parameters is more than 0.4, those two model-parameters are highly correlated to each other and the corresponding coefficients obtained through the multiple regression analysis may not be accurate. In this case study, coefficients of parameters $c_3$, $c_4$, $c_5$, $c_6$, $c_7$ and $c_8$ may not be very accurate. Fig.4.(a) and Fig.5.(a) show the fitting errors for N810 and Dev Phone, respectively. The results show that the fitting error of power consumption in N810 is only 0.29% on an average and 2.56% at the worst case. The errors for Dev Phone is only 1.39% on an average and 6.83% at the worst case. These show that the models of both N810 and Dev Phone are well fitted. However, this simply means that the models are well fitted to the power dissipated for training benches.

### D. Evaluation of Model Accuracy

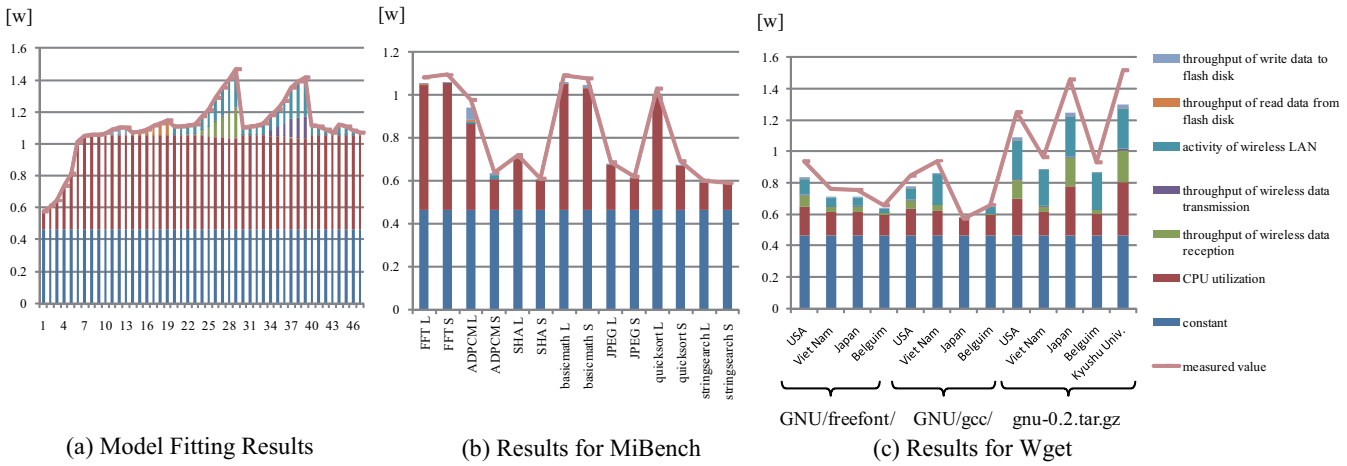
This section shows evaluation results for the power models. We compare measured and estimated power values for MiBench suite and Wget executed respectively on N810 and Dev Phone. The results of MiBench suite executed on N810 and Dev Phone are shown in Fig.4.(b) and Fig.5.(b), respectively. Line charts and bar charts respectively show the measured and estimated power consumption results. The average errors between estimated values and the measured values for N810 and Dev Phone are 1.96% and 2.06%, respectively. The accuracy is very good. Even for the worst case, the errors for N810 and Dev Phone are 3.95% and 4.37%, respectively. This good accuracy is because MiBench suite mainly activates CPU only. Although they slightly use a flash disk, they do not use wireless LAN at all. Therefore, the good results mean that coefficients related to the CPU utilization are very accurate. Unlike MiBench suite, Wget uses wireless LAN as well as the flash disk. Fig.4.(c) and Fig.5.(c) show the results of Wget executed on N810 and Dev Phone, respectively. The average errors for N810 and Dev Phone are 7.63% and 6.05%, respectively. These are worse than results for MiBench suite. However, the accuracy is still very good. In the worst case, the errors for N810 and Dev Phone are 14.46% and 15.18%, respectively. Compared to MiBench, the worst case error of Wget executed on Dev Phone is much worse. More specifically, the estimations of power dissipated for downloading gnu-0.2.tar.gz are inaccurate. One possible reason is a multicollinearity problem. Since there is a strong correlation among $P_{recep}$, $P_{trans}$ and $P_{wact}$ in training benches, corresponding coefficients may not be very accurate. To come up with a way to avoid the multicollinearity problem in the model fitting process is one of our future work.

### E. Overhead of Run-Time Power Analysis

Our power analysis method checks the status of a wireless LAN circuit every 200 milliseconds and collects the other parameter values every one hundred seconds. This involves a considerable power overhead. To evaluate the overhead, we run a program which periodically collects the values of the model-parameters on N810 and Dev Phone, respectively. The results demonstrate that the power overheads for collecting the parameter values are about 100mW and 46mW for N810 and Dev Phone, respectively. These are only 12% and 7% of the average powers respectively dissipated in the mobile terminals during MiBench suite is running. These are small. The important point is that those parameter values do not necessarily need to be collected very frequently. More specifically, the program for collecting the parameter values needs to run only when the power consumption needs to be analyzed. Therefore, the impact of the power analysis on a battery life of the mobile terminals is limited.

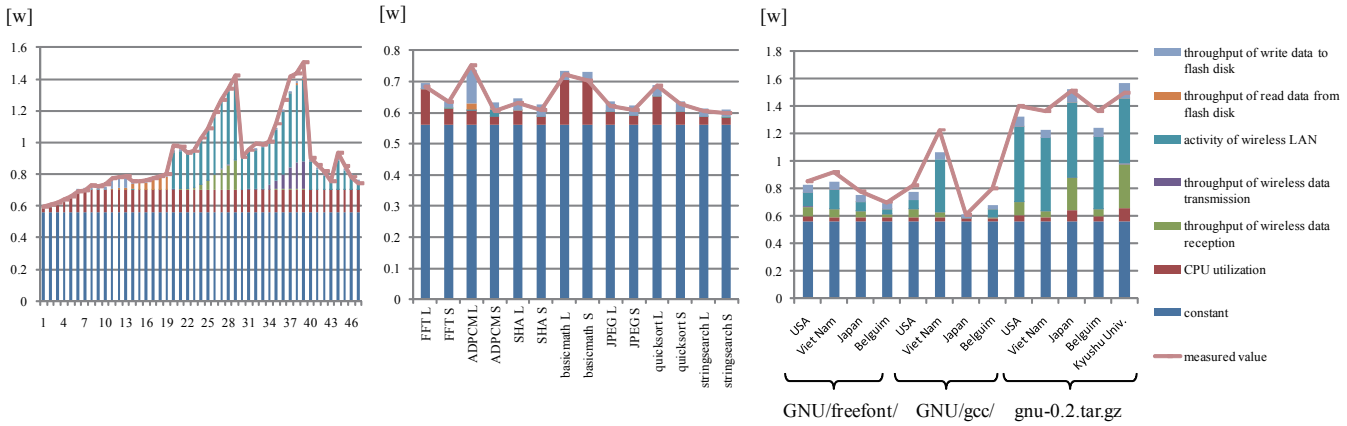
## V. SUMMARY

The paper presents a method to generate a lightweight power consumption model of wireless communication devices. Our model targets the power consumptions of not only CPU and memory but also entire mobile terminal including CPU, memory, flash disks, wireless LAN and other peripheral circuits. Once the model is generated for a given wireless terminal, the run-time power consumption of the wireless terminal can be estimated with a small number of parameters whose values can be quickly obtained through an OS. Experimental results with an N810 terminal developed by Nokia, and an Android Dev Phone developed by HTC demonstrate that the error of our model is on an average 4.33% compared to the measured power consumption. Since our approach estimates the power consumption using a small number of parameters which are inherently maintained by OS, the power consumption of application programs running on the device can be analyzed in real-time with a small power overhead. Our future work will be devoted to improve the model accuracy and to extend our current model to consider the power consumptions related to DSP, GPU and LCD monitor which includes a backlight. To come up with more practical applications of our model is also our future work.




(a) Model Fitting Results (b) Results for MiBench (c) Results for Wget

Fig. 4. Experimental Results with N810




(a) Model Fitting Results (b) Results for MiBench (c) Results for Wget

Fig. 5. Experimental Results with Dev Phone

## ACKNOWLEDGMENT

This work is partially supported by a Grant-in Aid for Young Scientists (B) (20700049).

## REFERENCES

[1] S. Kaxiras and P. Xekalakis, "4T-Decay Sensors: A New Class of Small, Fast, Robust, and Low-Power, Temperature/Leakage Sensors," in Proc. of ISLPED, pp.108-113, August 2004.

[2] W. L. Bircher, M. Valluri, J. Law, and L. K. John, "Runtime Identification of Microprocessor Energy Saving Opportunities", in Proc. of ISLPED, pp.275-280, August 2005.

[3] T. Li and L. K. John, "Run-time Modeling and Estimation of Operating System Power Consumption", in Proc. of Int'l Conference on Measurements and Modeling of Computer Systems, pp.160-171, June 2003.

[4] C. Isci and M. Martonosi, "Runtime Power Monitoring in High-end Processors: Methodology and Empirical Data," in Proc. of Internatinal Symposium on Microarchitecture, pp.93-104, 2003.

[5] W. Wu, L. Jin, J. Yang, P. Liu, and S. X.-D. Tan, "A Systematic Method for Functional Unit Power Estimation in Microprocessors," in Proc. of DAC, pp.554-557, July 2006.

[6] K. Singh, M. Bhadhauria, and S.A. McKee, "Real Time Power Estimation and Thread Scheduling via Performance Counters," in Proc. of workshop on Design, Architecture and Simulation of Chip Multi-Processors, Nov. 2008.

[7] W. L. Bircher and L. K. John, "Complete System Power Estimation: A Trickle-Down Approach Based on Performance Events," in Proc. of ISPASS, pp.158-168, April 2007.

[8] P. Bellasi, W. Fornaciari, and D. Siorpaes, "Predictive Models for Multimedia Applications Power Consumption based on Use-Case and OS Level Analysis," in Proc. of DATE, pp.1446-1451, April 2009.

[9] D. Lee, T. Ishihara, M. Muroyama, H. Yasuura, and F. Fallah, "An Energy Characterization Framework for Software-Based Embedded Systems," in Proc. of IEEE Workshop on ESTIMedia, pp.59-64, Oct. 2006.

[10] W. Ye, N. Vijaykrishnan, M. Kandemir and M.J. Irwin, "The Design and Use of SimplePower: A Cycle-Accurate Energy Estimation Tool", Proc. of 37th DAC, pp.340-345, June 2000.

[11] D. Brooks, V. Tiwari, and M. Matonosi, "Wattch: A Framework for Architectural-Level Power Analysis and Optimization", in Proc. of ISCA, pp.83-94, June 2000.

[12] V. Tiwari, S. Malik, and A. Wolfe, "Power Analysis of Embedded Software: A First Step towards Software Power Minimization", in Proc. of ICCAD, pp.384-390, November 1994.

[13] C. Brandolese, W. Fornaciari, F. Salice, and D. Sciuto, "An Instruction-level Functionality-based Energy Eestimation Model for 32-bit Microprocessors," in Proc. of DAC, pp.346-351, June 2000.

[14] T. Sinha, and A. P. Chandrakasan, "JouleTrack - A Web Based Tool for Software Energy Profiling", in Proc. of DAC, pp.220-205, June 2001.

[15] A. Muttreja, A. Raghunathan, S. Ravi, and N. K. Jha, "Hybrid simulation for embedded software energy estimation," proc. of DAC, pp.23-26, June 2005.

[16] PowerTop by Lesswatts.org, http://www.lesswatts.org /projects/powertop

[17] T. K. Tan, A. Raghunathan, G. Lakshminarayana, and N. K. Jha, "High-level Software Energy Macro-modeling", in Proc. of DAC, pp.605-610, June 2001.

[18] F. E. Harrell, "Regression Modeling Strategies," Springer-Verlag, 2001.