A Real-Time Power Analysis Platform for Power-Aware Embedded System Development

LIANG-BI CHEN, YEN-LING CHEN AND ING-JER HUANG
Department of Computer Science and Engineering
National Sun Yat-sen University
Kaohsiung, 804 Taiwan
E-mail: {liangbi@eslab.; ijhuang@}cse.nsysu.edu.tw

This paper proposes a real-time power analysis platform, including both hardware and software modules, which is capable of profiling, analyzing and controlling power behavior for power-efficient/aware embedded system applications that can be used for both educational and research purposes. The platform consists of an SoC development board with power measurement probes and supporting power measurement mechanisms, application phase tagging, power analysis, and power control. It allows the developer to profile and analyze the real-time power usage of various major parts of a major hardware system while running an embedded system application on the development board. Application phase tagging makes it possible for the developer to associate the power usage with the application behavior in detail, and the power control mechanism makes it possible to develop power-aware applications capable of dynamically adjusting power behavior while sustaining necessary performance. A digital still camera (DSC) was used as a case study to test this platform. We first characterized the power usage of the DSC and then improved its power efficiency with power-aware operations.

Keywords: embedded systems, energy-aware, low-power design, power analysis, power-aware, power control, power management, power measurement, power optimization

1. INTRODUCTION

With the increasing popularity of portable devices such as cellular phones, video recorders/players, digital still cameras (DSCs), and personal digital assistants (PDAs), etc., design methodologies which help the design, analysis and optimization of power efficient/aware embedded applications have become a necessity in education, research and production development. Simulation related approaches are widely adopted to achieve such purposes. The power or energy consumed by an embedded application is estimated by calculating the energy consumption of various hardware components in the target circuits through simulations that can be executed at different levels of abstraction such as circuit, logic gate, RTL, or behavior levels. One limitation of simulation-based techniques is that they might not be able to faithfully model real hardware behavior. There may be too many components to be modeled, or the corresponding models simply are not available or compatible with each other. In addition, it is difficult to analyze complex applications due to the slow simulation time. Furthermore, real-time behavior cannot be captured in the simulation environment.

These shortcomings can be avoided by measuring and analyzing the power con-
sumption directly on real hardware. However, most of the commercially available development boards for embedded systems (such as ARM’s Versatile platform development board [2], LogicXtool [3], Philips LPC2000 [4], and PIC Development Kit [5], etc.) do not provide any power related mechanism for this purpose.

Motivated by the above needs, we developed in cooperation with Microtime Computer Inc., a power analysis platform consisting of an ARM-based SoC development board and supporting power-related hardware and software modules [6]. This platform makes it possible for a developer to measure, analyze and control the power behavior of complex embedded applications in real time and may helpful to the education, research, and product development of low-power embedded systems.

The rest of the paper is organized as follows. Section 2 reviews the related work. Our proposed platform is introduced in section 3. A case study demonstrating the use of the proposed platform is described in section 4. Finally, we conclude in section 5.

2. RELATED WORK

Previously published works for power measurement and monitoring focus predominantly on instruction-level power dissipation and on simulation models rather than on hardware measurement. The idea of developing an instruction-level power model for individual processors was proposed by Tiwari et al. [7]. In that study, power consumption was modeled as a base cost for each instruction plus circuit state overhead that depended on neighboring instructions. Chang et al. [8] and Lee et al. [9] proposed an energy instruction model for the analysis of RISC and ARM7TDMI processors, respectively. Contreras et al. [10] proposed a high-level power simulator tailored for the Intel XScale Processor called XTREM.

These simulators estimate the power consumption of microprocessor-based systems more quickly and at lower cost than measurement-based techniques. However, these tools have low estimation accuracy, which makes meaningful energy tuning difficult. For example, Ghiasi and Grunwald [11] showed that different energy simulators could produce contradictory results for the same application running on the hardware platform. Furthermore, simulation-based techniques might not be able to faithfully model the real hardware behavior. There may be too many components to be modeled, or the corresponding models simply are not available or incompatible with each other. In addition, because simulation time is long, analyzing complex applications is difficult. However, one way to avoid these problems is to measure the power consumption directly on real hardware.

Several approaches have been used to measure power consumption of electrical equipment. Simunic et al. [12] evaluated power consumption of the ARM7 processor using milliampere meter. Russell et al. [13] measured power consumption of a processor using a register to record measurements taken at regular intervals which was then used to calculate the mean consumption. Flinn et al. [14] developed a tool they called PowerScope to profile the energy used by a mobile application. The PowerScope can analyze energy consumption based on the power used by various segments of a program structure. By providing such detailed feedback, attention can be focused on target system components responsible for using most of the energy. Isci and Martonosi [15] combined total
power values and power used per cycle to estimate power used per unit estimation by Intel Pentium 4 processors. Tiwari et al. [16] used a digital multimeter to physically measure energy consumption of three commercial processors based on the instruction energy model, becoming the first to apply physical measurement of overall power consumption of a digital system. Knostantakis et al. [17] proposed an approach to measuring power consumption of digital circuits and embedded systems, an approach based on measure of power used by various configurations over different time intervals without being affected by the clock frequency of a digital circuit.

The type of hardware used in an embedded system is relevant to the minimization of power consumption because an embedded system consists not only of a processor but also of many peripheral units, which heavily contribute to the power consumption of the whole system. In addition, the contribution of software to the total power consumed during execution is becoming increasing important in portable embedded systems where power is a constraint. Therefore, accurate measurement and analysis of power consumption are essential for the evaluation of the software/hardware related power consumption of a processing system and its peripheral units, which is important for analyzing total power budgeting and distribution which is can be used to make early decisions such as hardware/software portioning.

In order to ensure that the power consumption is minimized, there is a need for the accurate measurement and analysis of power dissipation for each component of the whole system, which can be difficult when measuring power used by peripherals in non-prediction applications such as audio’s A/D conversion and games, etc. In addition, most studies have focused on the power used to run entire application programs, not separate phases of those programs, which would provide data useful for designing power-aware embedded systems. To analyze the separate phases in such applications, some kind of phase tagging is needed. Power-aware embedded systems should also include a power control mechanism for efficient power conservation, which would, for example, extend battery-life. Such a platform would include a workload monitoring function to help implement aggressive power control strategies similar to that by Benini et al. [18].

However, while most published papers focus on measuring and analyzing power consumption of the processor, they often disregard the peripherals. In order to ensure that the power consumption is minimized, there is a need for the accurate measurement and analysis of power dissipation that power used by both the processing system and its peripheral units. This has been performed accurately with NI Labview and DAQ tool chain [19], but this method lacks a phase tagging function and thus cannot provide detailed analysis. In addition, the cost of NI Labview and DAQ tool chain prohibits its use where many sets are required such as in classes where many students need to analyze power usage of embedded systems.

In this paper, we describe a platform we develop that can be used to measure, analyze, and control power usage hardware and software used in a processing system and its peripheral units. The platform consisting of an SoC development board with power measurement probes and the supporting mechanisms for power measurement, application phase tagging, power analysis, and power control, allows the developer to profile and analyze the real-time power usage of each major hardware part when running an embedded system application. With application phase tagging and power control mechanism, power usage can be analyzed by application behavior in more detail and power-aware
applications capable of dynamically adjusting power behavior while sustaining necessary performance can be developed.

3. PROPOSED PLATFORM FOR REAL-TIME POWER ANALYSIS

The block diagram and the photograph of the proposed platform are shown in Figs. 1 and 2, respectively. The platform consists of an ARM-based SoC development board (Fig. 1 (a)), power measurement module (Fig. 1 (b)), application phase tagging module (Fig. 1 (a)), and software modules (Fig. 1 (c)). These modules are described below.

![Block diagram of the proposed real-time power analysis platform for embedded system developments; (a) An SoC development board with an FPGA device; (b) The power measurement hardware module; (c) The PC-based host with USB I/O interface.](image)

![The photograph of all hardware components of the proposed platform.](image)
3.1 The ARM-based SoC Development Board (PreSoC) with Power Measurement Probes

We have cooperated with Microtime Computer Inc. to define an ARM-based SoC development board, called PreSoC. The development board consists of a motherboard which hosts most of the I/O peripherals (such as SDRAM, FLASH memory, CF card reader, LCD, CMOS sensor, UART, audio codec, Ethernet, and USB, etc.), two changeable daughter boards (an ARM7 CPU and an FPGA device), an in-circuit emulator (ICE) and a host debugger [6]. Fig. 3 is the photograph of the development board.

A unique feature of the development board is that it provides measurement probes to measure the electric currents of eleven major devices: CPU, Ethernet, UART, SDRAM, expansion memory, FLASH memory, LCD, CF Card reader, CMOS sensor, audio codec, and USB. These probes are connected to the power measurement hardware module, presented in the next section, to obtain the power information of each device.

![Fig. 3. The photograph of the development board and its power measurement probes.](image)

3.2 Power Measurement

Power measurement is conducted with a hardware module and a software module. The power measurement hardware module consists of current-to-voltage circuits, differential circuits, voltage amplifiers, and a multi-channel A/D converter, as shown in Fig. 1 (b). The power measurement software module runs on a PC-based host.

3.2.1 Power measurement hardware module

The power measurement hardware module in Fig. 1 (b) has four components: the current-to-voltage circuit, the differential circuit, voltage amplifier, and the A/D conver-
The current-to-voltage circuit converts the electric current from a measurement probe on the development board to a voltage because the A/D converter can only process the voltage. Because the electric currents from different measurement probes may differ significantly, a DIP switch is used to select a suitable resistance value for a corresponding measurement probe. The next circuit is the differential circuit. The circuit has an operation amplifier (OPA) and some resistors. The differential circuit is used to avoid potential short circuit problems, as illustrated in Fig. 4. On the development board, the measurement probe of each device is located near the power supply end. If the outputs of the current-to-voltage circuits are directly connected to an A/D converter which is capable of simultaneously measuring multiple channels (i.e., the results of multiple measurement probes), then the same ends of the measurement probes would be shortened together (e.g., \( V_2 \) and \( V_4 \) in Fig. 4). Normally, a differential circuit would be required to isolate these ends. However, this circuit is not needed when measurement probes are inserted between the target device and the GND end.

![Fig. 4. PreSOC SoC-based embedded systems development board’s measurement probes.](image)

The next circuit is the voltage amplifier, which is used to amplify the output signals of the differential circuits for more accurate measurement by the A/D converter. The voltage amplifier consists of an OPA, some resistors and a DIP switch to select one of the resistors for a desired amplification scale. This configurable amplification scale is necessary because the electric currents from different measurement probes may differ significantly (e.g., the electric currents of the CPU and the FLASH memory on the development board are 130mA and 30mA, respectively).

The next circuit is a multi-channel A/D converter which converts analog signals to digital signals. In our current design, the HT46R23 A/D-type 8-bit microcontroller [22] is used as the multi-channel A/D converter. This microcontroller provides a programmable 8-channel 10-bit A/D converter with 8 MS/s maximal sample rate.

3.2.2 Power measurement software module

The power measurement software, running on the host PC, configures the multi-
channel A/D converter through a USB-based I/O interface. As can be seen in Fig. 5, the configuration screen, the developer could assign the name of the device being measured and set its amplification scale, working voltage, enable or disable usage, and set a specific color that will be used to display results for each channel (Fig. 6). The software can also be used to adjust the sampling rates of the measurement probes, as shown in the upper right corner of Fig. 6.

The power measurement software reads the digital values from the multi-channel A/D converter. In addition, the power measurement software also reads the application phase tagging information, introduced below in section 3.3. Another software (power analysis software module), presented later in section 3.4, constructs the power information from these digital values and associates them with the application phase tags.

![Fig. 5. The channel configuration screen of the power measurement software.](image)

![Fig. 6. Current/power/energy trace of each device, annotated with application phase tagging signals.](image)

3.3 Application Phase Tagging

The real-time power information (trace) generated by the power measurement hardware module would be more helpful if it could be associated with the behavior of the embedded application. To achieve this goal, an application phase tagging port was implemented in the FPGA device on the development board. This port can be accessed by the development board through the memory-mapped I/O mechanism (at the memory ad-
address 0x03F0C004 in our current implementation). The application running on the development board can write an identification tag to the application phase tagging port, with a memory write operation, to indicate the particular moment when a program is executed.

On the other hand, the application phase tagging port can be accessed by the external world from the I/O pins on the FPGA daughterboard. The power measurement software reads the application tags with a USB I/O interface at the same time while reading the digital voltage values from the power measurement hardware module.

3.4 Power Analysis

The power analysis software running on the host PC combines the voltage values and the application phase tags, which are both collected by the power measurement software, to produce meaningful power analysis results and displayed on a sophisticated graphical user interface (GUI) (Figs. 6 and 8).

As can be seen in Figs. 1 (b) and 4, current-to-voltage circuits are used to probe the power supply voltage end, namely $V_1$ ($V_3$). To measure the instantaneous power, an absolute measurement of the voltage at the target device, namely $V_2$ ($V_4$), is multiplied by the difference in voltage across a resistor (($V_1 - V_2$) or ($V_3 - V_4$)), and scaled according to the resistance, $R$, as follows,

$$P(t) = I(t) \cdot V(t) = \frac{(V_1(t) - V_2(t)) \cdot V_1(t)}{R}$$

(1)

For calculation of the value of the electric current, the value of $R$ is 1Ω. Accordingly, when $V = I \times R$ and $R = 1$, then the value of $V$ is equal to that of $I$.

Fig. 6 shows one menu of the GUI. The traces of electric current, power, and consumed energy for the CPU, SDRAM, FLASH memory and CMOS sensor are displayed on the screen. These traces are measured for the implementation/development/installment of software for a digital still camera (DSC) place on the development board (details to be presented in section 4).

The DSC application writes out six application phase tags to indicate the states (such as selecting the pixel size, performing the JPEG compression, etc.) of the DSC application as shown on the lower right corner of the figure. These application states correspond to the vertical cut lines in the current/power/energy traces, associating the phases of DSC algorithm to the variations and durations of these traces.

The power analysis software also analyzes the maximum/minimum/average current/power/energy for each device, as shown in Fig. 9. With such data, developers can identify the devices that consume the most energy and design power-aware embedded applications.

3.5 Power Control

The power control function is emulated by a software module. The power control software module operates in two procedures. The first procedure is to configure the battery information analyzer and the power mode switch. The second procedure is to emu-
late the functionality of the battery information analyzer and the power mode switch and send the power control signal directly to the power mode configuration port. The second procedure is especially useful for educational experiments or during the development stage of the embedded applications. In such cases, it is no necessary or feasible to wait for a “real” battery to deplete its capacity in order to experiment with different power situations. We can simply use this software to emulate the battery’s behavior and then directly control the embedded application running on the development board. Fig. 7 illustrates the power control software running at the emulation mode with the battery of 5.234% capacity.

![Emulation of battery capacity and the power control.](image)

In order to acquire real battery information, the battery information analyzer can be implemented by using a battery management chipset such as TI bq2084, TI bq29312, or TI bq29400 [23] to obtain the battery information. The power mode switch generates a power control signal to the power mode configuration port based on the information obtained from the battery information analyzer. Similar to the application phase tagging port, the power mode configuration port is implemented in the FPGA device and is connected to the power mode switch through the I/O pins on the FPGA daughterboard. The application running on the development board could read the power control signal through the power mode configuration port with a memory read operation. The application can then adjust its behavior according to the power control signal, turning the application into a power-aware one.

4. CASE STUDY: POWER CHARACTERIZATION AND OPTIMAZATION OF A DIGITAL STILL CAMERA

In this case study, we demonstrate how the proposed platform can be used to evaluate
the power consumption of a software implementation of the digital still camera (DSC),
the power variation of different DSC configurations, and how the information provided
by the platform can be used to turn the DSC into a power-aware one. The DSC is a popu-
lar consumer electronic device that can capture images (“take pictures”) and store them
in digital format [24]. If we want to develop a power-aware DSC that relies on a battery
with limited capacity, we must consider the tradeoff between quality, performance and
power consumption, a common problem encountered in the development of many em-
bedded applications.

The configuration of the development board was: Samsung S4510B microcontroller
chip with an ARM7TDMI CPU core, a 2 Mbytes FLASH memory, a 16 Mbytes SDRAM,
a CMOS senor, a 4 × 4 keypad, and a 128 × 128 LCD display with 4 gray level. The
S4510B chip runs at 25 MHz. The experiment was conducted to take one picture and
save it to a remote PC through the UART (COM port).

4.1 Analysis of Power Traces and Device Characteristics

The current/power/energy traces of the DSC are shown in Fig. 6. The power trace is
magnified in Fig. 8. There are five application phase tags dividing the power trace into
six phases: load program, start program, set pixel size, take picture, execute JPEG com-
pression, and save to UART. There are several observations that we can derive from the
traces.

(1) The power of the FLASH memory is consumed only during the initiation phase at
time the program is loaded from the FLASH memory to the SDRAM. After that,
the FLASH memory consumes very little power.

(2) The power traces of the CPU and SDRAM fluctuate a lot except during the JPEG
compression phase. The JPEG compression requries very heavy computation and
thus the CPU and SDRAM are busy all the time and thus sustain a constantly high
power consumption during such phases. On the other hand, the loading of the rest of
the phases are relatively light. Therefore, the CPU and the SDRAM are not busy all
the time and thus the power traces fluctuate.

(3) The power trace of the CMOS sensor remains almost constant throughout all the
phases. This is due to the fact that the CMOS sensor module used in this experiment
does not provide a turn-off mode and thus is active all the time.

Fig. 8. Magnified depiction of the power consumption of a program actually executed by the DSC.
Fig. 9. Power consumption analysis of each device.

Fig. 9 shows the detailed analysis of power consumption for each device. As can be seen in the pie chart in the figure, the CPU consumes 55.68% of the power, and the SDRAM, FLASH memory and CMOS sensor 22.2%, 1.04%, and 21.09%, respectively. This analysis suggests that optimization on the DSC algorithm or operation modes could reduce the power consumption of the CPU and SDRAM, and because they represent a great portion of total power consumption, the total power consumption could be decreased significantly. A CMOS sensor could be turned off when not in use, also providing significant power conservation.

Fig. 9 also lists the minimum, maximum and average current and power for each device on the development board. Because the numbers in the figure might not be clear, we list the power information of the four devices and study their variations in Table 1.

<table>
<thead>
<tr>
<th>Power variation of the four devices.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device</td>
</tr>
<tr>
<td>CPU</td>
</tr>
<tr>
<td>CMOS Sensor</td>
</tr>
<tr>
<td>FLASH</td>
</tr>
<tr>
<td>SDRAM</td>
</tr>
</tbody>
</table>

4.2 Turning the DSC into Power Awareness: Trading off Picture Quality, Hardware Operation, and Battery-Life

With the basic analysis described in the previous section, we are now ready to explore several interesting features in order to develop a power-aware DSC. The features
that we explored are listed in Tables 2 and 3, respectively. Table 2 lists the optimization techniques for the Discrete Cosine Transform (DCT), which is the major computation involved in JPEG compression [25]. Our DSC is implemented by a one-dimension DCT. The DCT formula [26] is given as the following,

\[ F(u) = \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} \cos \left( \frac{(2i+1)\pi}{16} \right) f(i) \]

where \( C_u = 1/\sqrt{2}; u = 0; C_u = 1; \) any other. The direct implementation of this equation needs 64 multiplications. Because the cosine has cyclic properties and is symmetric, the number of multiplications can be reduced.

In Option 0, the original DCT algorithm looks up the cosine value from a table and performs the calculation with floating-point numbers. This takes a lot of time and leads to an extremely inefficient operation. There are many options to optimize the DCT.

In Option 1, we could perform the calculation with fixed point numbers instead of floating point numbers at the cost of sacrificing accuracy. In Option 2, we could determine the cosine values beforehand since the angles in the cosine function are limited to some certain degree. In Option 3, we unroll some of the important “for loop” in the algorithm.

| Table 2. DCT optimization techniques in the JPEG algorithm. |
|--------------|-------------------------------------------------|
| Option | Optimization Technique                     |
| 0        | Cosine table lookup and floating point operations (used in the original DCT algorithm) |
| 1        | Fixed point operations                      |
| 2        | Pre-calculated Cosine values                |
| 3        | Loop unrolling                              |

Table 3 lists the options of conserving power at the expense of picture quality and hardware function. There are two parameters which affect the picture quality: the picture size and the possible color filter array (CFA) demosaicing algorithms which post process the pixel values read from the CMOS sensor. Here we choose two picture sizes and three demosaicing algorithms. To save hardware further power, we choose to turn off the LCD when it is not in the preview mode.

Fig. 10 compares the energy consumption per picture for all the options in Tables 2 and 3. The observations are as follows,
1. The options in Table 2 are very helpful in reducing the energy consumption and can be applied accumulatively. For each picture size, the “+” sign in the picture indicates that all the options to its left side are included in the measurement. When all are taken, they can reduce the energy consumption to 1/10 or even less.

2. The “smooth hue-transition interpolation” CFA demosaicing algorithm produces the best picture quality but consumes the most energy while the “nearest point copied” algorithm consumes the least energy while giving worst quality. The “bilinear interpolation” algorithm falls in the middle in both picture quality and energy consumption.

3. As anticipated, the energy consumption is roughly proportional to picture size in pixels.

Based on the above analysis, we conclude that the options in Table 2 should all be used to optimize DCT, and that the combinations of options in Table 3 should remain since each combination gives a unique energy-quality tradeoff. Therefore, the final structure of the DSC is illustrated in Fig. 11, with two alternatives in the picture size and three alternatives in the CFA demosaicing algorithm.

We further compared the battery capacity and battery life (represented as the number of pictures taken for a given battery capacity) for each possible operation combinations (as in Fig. 11) and results are shown in Fig. 12. The Y-axis of the figure represents remaining battery capacity. Assuming that a full battery contains 7000 Joules of energy and the X-axis is the number of pictures taken, there are six possible operation combinations.
Fig. 11. The operation alternatives of the power-aware DSC.

Fig. 12. Number of pictures taken: the power-aware DSC vs. the six pixel-CFA demosaicing algorithm combinations.

(Fig. 11). Their energy-pictures relationships are the six straight lines in Fig. 12. The (320 × 240 pixels, smooth hue-transition) combination, the straight line furthest left in the figure gives the best picture quality but produces the fewest number of taken pictures (230). On the other hand, the (160 × 120 pixels, nearest point copied) combination represented by the straight line furthest right in the figure gives the worst picture quality but produces the greatest number of taken pictures (930).

Based on such analysis, we could imagine a smart power-aware DSC which starts from taking good quality pictures when the battery is full; however, as the battery's remaining capacity decreases to a certain level, the DSC switches to a mode taking picture with less quality to maximize the number of pictures that can be taken given the remain amount of power in the battery. Table 4 defines four operation modes for this power-aware DSC. It starts at Mode 1 when the battery is full, and then moves to the next mode when the battery capacity becomes 25% less.
Table 4. Operation modes of the power-aware DSC.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Picture Size (pixels)</th>
<th>CFA Demosaicing Algorithm</th>
<th>Turn off Device</th>
<th>Picture Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>320 × 240</td>
<td>Smooth hue-transition interpolation</td>
<td>–</td>
<td>best</td>
</tr>
<tr>
<td>2</td>
<td>320 × 240</td>
<td>Bilinear interpolation</td>
<td>–</td>
<td>good</td>
</tr>
<tr>
<td>3</td>
<td>160 × 120</td>
<td>Smooth hue-transition interpolation</td>
<td>–</td>
<td>poor</td>
</tr>
<tr>
<td>4</td>
<td>160 × 120</td>
<td>Nearest point copied</td>
<td>LCD</td>
<td>worst</td>
</tr>
</tbody>
</table>

The result of the power-aware DSC is shown by the segmented thick line. As the battery capacity drops, the mode of the DSC changes to trade picture quality for longer battery life (more pictures to be taken) as the slope of the line becomes less steep. With such an arrangement, a total of 590 pictures can be taken, with 60, 70, 220, 240 being in Mode 1 through 4, respectively. This keeps a better balance between the total number of taken pictures vs. the picture quality. The power-aware DSC is found to use energy much more efficiently when it is able to take both quality and energy into consideration.

5. CONCLUSION

We have proposed a real-time power analysis platform which is capable of profiling, analyzing and controlling the power behavior for developing a power-efficient/aware embedded system application for both educational and research purposes. This platform consists of an SoC development board with power measurement probes and the supporting mechanisms for power measurement, application phase tagging, power analysis, and power control. This platform allows the developer to profile and analyze the real-time power usage of each major hardware by the embedded application. With the help of the application phase tagging, the developer can accurately associate the power usage with the application behavior. In addition, the power control mechanism allows the development of power-aware applications capable of dynamically adjusting power behavior while sustaining necessary performance and quality. We have successfully demonstrated the usefulness of this platform with a case study by first characterizing the power usage of a DSC and then improving its power efficiency with through dynamically taking quality and energy into consideration. The future directions of this work include the design of a development board supporting multiple voltages and frequencies and the corresponding power control mechanisms and policies.

ACKNOWLEDGEMENTS

The authors would like to thank Microtime Computer Inc. for their generous financial and technical supports, and thank Mr. James Steed and Ms. Jinrui Huang for proofreading the draft.

REFERENCES

vest.htm.
A REAL-TIME POWER ANALYSIS PLATFORM

design experiments,” ACM Transactions on Design Automation of Electronic Sys-
sis for platform-based design of multimedia applications,” ACM Transactions on
24. F. Vahid and T. Givargis, Embedded System Design – A Unified Hardware/Software
26. T. Masaki, Y. Morimoto, T. Onoye, and I. Shirakawa, “VLSI implementation of
inverse discrete cosine transformer and motion compensator for MPEG2 HDTV
video decoding,” IEEE Transactions on Circuits and Systems for Video Technology,

Liang-Bi Chen (陳夷弼) received the B.S. and M.S. degrees in Electronic Engineering from the National Kaohsiung University of Applied Sciences, Kaohsiung, Taiwan, R.O.C., in 2001 and 2003, respectively, and is currently working toward the Ph.D. degree in Computer Science and Engineering at the National Sun Yat-sen University, Kaohsiung, Taiwan, R.O.C. From August 2008 to September 2008, he had an internship in Department of Computer Science at the National University of Singapore, Singapore. He was a visiting researcher in Department of Computer Science at the University of California, Irvine, CA, U.S.A. between September 2008 and August 2009. His research interests are VLSI design, power/performance analysis for embedded mobile applications and devices, power-aware embedded systems design, low-power systems design, and SoC debugging. He is a student member of the ACM, IEEE, IEICE, and Taiwan IC design Society.

Yen-Ling Chen (陳姵伶) completed the work in this paper while completing the B.S. degree in Computer Science and Engineering from the National Sun Yat-sen University, Kaohsiung, Taiwan, R.O.C. Her research interests include power/performance analysis for power-aware embedded systems design, and wireless network protocol design/analysis. She is currently working toward the M.S. degree in Computer and Communication Engineering at the National Cheng Kung University, Tainan, Taiwan, R.O.C.
Ing-Jer Huang (黃英哲) received the B.S. degree in Electrical Engineering from the National Taiwan University, Taipei, Taiwan, R.O.C., in 1986, and the M.S. and Ph.D. degrees in Computer Engineering from the University of Southern California, U.S.A., in 1989 and 1994, respectively. He is currently a Professor in the Department of Computer Science and Engineering at the National Sun Yat-sen University, Kaohsiung, Taiwan, R.O.C. His research interests include computer architecture, SoC design, design automation, system software, embedded systems, and hardware/software co-design/verification. He has actively involved in academic, educational and industrial activities. He has extensive collaboration with several IC related industries. He is a member of the IEEE and ACM.