EPC: A Power Instrumentation Controller for Embedded Applications

Ioannis Manousakis
Foundation for Research Technology Greece
Nikolaou Plastira 100, Vassilika Vouton
P.O Box 1385. GR-70013
Heraklion, Crete, Greece
jmanous@ics.forth.gr

Dimitrios S. Nikolopoulos
School of Electronics, Electrical Engineering and Computer Science
Queen's University of Belfast
CBT9 5BN, Belfast, UK
d.nikolopoulos@qub.ac.uk

ABSTRACT

In this work we propose and implement a real-time power monitor controller based on a simple 8-bit AVR controller and an analog Hall effect current sensor. Our setup imposes negligible degradation in the power efficiency, performance and the response time of the instrumented system. Those characteristics make it ideal for portable and battery critical applications. The use of an external controller enables the implementation of a function set for automated power measurement and energy accounting. In order to validate the correctness and the quality of our implementation we have used our setup to instrument a Linux Single Board Computer (SBC) based on an ARM micro-controller. During this instrumentation we have run various CPU and I/O intensive workloads that incur fast phase transitions.

Categories and Subject Descriptors

H.4 [**Embedded Systems, Power, Energy**]: Miscellaneous; D.2.8 [**Linux**]: Instrumentation—*Power, Energy*

General Terms

Energy, Power, Embedded Systems, Controller, Single Board Computer

Keywords

Power, Energy, Embedded, Controller, Single Board Computer, Hall Effect, Embedded Linux

1. INTRODUCTION

Power dissipation of electronic devices has emerged as the leading problem in computer architecture. Thus, there has been a lot of effort in designing energy efficient systems with self power down capabilities. Typically, those include techniques for complete component shutdown, Dynamic Frequency Scaling (DFS) or Dynamic Voltage Scaling (DVS). Those tools are valuable, for example if the target is to stay below the Thermal Design Power (TDP) of a device or to discover on the fly the most energy efficient operating frequency for a specific workload.

Taking critical decisions about the system's operating frequency or voltage requires knowledge about it's instant power consumption. Lack of such information leads to complex static models and non realistic assumptions that ignore current system state and specifications [5, 3]. In order to overcome this limitation, current-sense devices can be installed to instrument and expose instant power consumption to the user environment. Traditionally, current sensors use a respectful amount of resistance that dissipates valuable energy and decreases the power efficiency of the system. In portable and battery-critical applications this side effect can effectively void the benefits of a run-time power analysis.

In this work we design and evaluate EPC, a power instrumentation controller that balances between data quality, cost, complexity and ease of use. Unlikely most traditional current-sense schemes, EPC features a *Hall effect* current sensor that dissipates zero energy during operation. This unique feature makes our approach ideal for battery critical embedded applications. In addition to EPC presentation, we discuss how self instrumented Linux systems can utilize real time, fine grained power consumption information to build energy optimization tools that exceed by far the granularity of the Linux scheduler.

2. CURRENT SENSE TECHNIQUES

There are many ways to measure current when designing the PCB. Those split into two categories *i) before and ii) after or onto the Voltage Regulator Module (VRM).* In [4], some techniques of the *second category* are featured and cross compared. All of them trade between energy dissipation, cost, complexity and bandwidth. Current indication is a valuable tool for the VRM designer, especially for overcurrent protection. Potentially, it may also be useful for real time power monitoring although it is rarely (or never) exposed. This scenario would be ideal for the system designer since he could obtain a per-component power consumption breakdown of the system (MCU, RAM, Miscellaneous I/O chips). Despite the advantages of the above method, cost is the primary concern that currently prevents such fine grained implementations.

In many cases, it is desirable to add self-power monitoring capabilities to an existing system. We consider this scenario as the common case today, as there are not many systems providing them. Ideally, we would like to interfere between the VRMs and the units of interest as we described before. Even if the cost, the design and the implementation are affordable, disturbing the delicate electrical characteristics of the modules, by inserting a parasitic circuit, may result into

a non-functional unit. In addition, even soldering onto the PCB might prove to be very hard. Those limitations categorize such an approach as not practical.

Respecting the previous restrictions, we are forced to place a current-sense device *before* the VRM practically outside the PCB and directly on the main power supply line. Figure 1 presents the typical current instrumentation setup for an embedded system. A power supply (battery, DC-DC or AC-DC step down converter) supplies with constant voltage the system from a single power line. This line is interrupted and measured by a current sensor (Section 2.1) and then heads into the VRMs and the power planes of the PCB. An output voltage is produced by the current-sense device that is proportional to the line current. This voltage is amplified, converted into digital values and stored into a data acquisition system. Since the analog signal is amplified with a non ideal op-amp, it is a common practice to filter it either with an analog (an RCL Low pass filter) or a digital method (Fast Fourier Transformation).

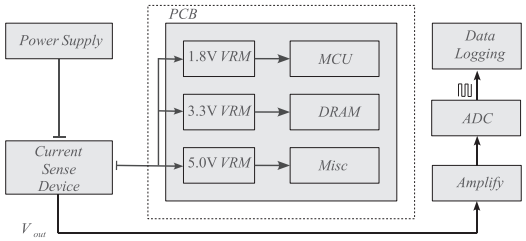

Figure 1: Typical setup for open loop current instrumentation in embedded applications.

Current sensors are analog devices that alter their output proportionally to an input current change. They split into two main categories *i) direct and ii) indirect*. Direct sensors typically consist of a resistor placed in series with the load in order to observe the voltage drop across it. From Ohm's Law, this drop translates directly to the instant current that flows through the resistor. On the other hand, indirect sensors do not feature any parasitic resistance, but calculate the current flowing through the conductor from side effects such as the magnetic field that is generated.

2.1 In series Resistor

Placing a resistor R_{Sense} in series with the load *(direct)* is the most common practice for current instrumentation. This technique has the advantages of being cost effective and simple. On the other hand it dissipates energy and converts it into heat. This can lead into a significant degradation of power and thermal efficiency of the device (typically 5-10%) resulting into shorten battery life or more expensive power supply unit. Ultimately any gain achieved from live power monitoring can be canceled. The obvious solution to lower the R_{Sense} value in order to waste less energy, has also negative side effects. Since the produced value is analog, placing an smaller resistor will cause a more narrow reference voltage (V_{Out}) range. In order to convert V_{Out} into digital values, either a more precise Analog To Digital (ADC) converter or a dedicated current-sense amplifier can be used, both with increased cost.

2.2 Hall effect

Hall effect current sensors *(indirect)* consist of a transducer that converts the magnetic field produced by the primary conductor (carrying the main load) into voltage. This reference voltage (V_{Out}) is proportional to the input current. Their main advantages over the traditional in series resistor sensors are *i) very low resistance (as low as the internal resistance of a wire), ii) linearity across the operating range and iii) immunity to humidity, vibrations and pressure*. All the above make this sensor class ideal for portable, energy critical applications. Until recently though, this sensor class was focused on high currents (greater than 10A) making them useless in embedded systems. A migration into a compact, cheap, low current chip with integrated amplifier enabled their expansion into portable devices.

3. DATA ACQUISITION TECHNIQUES

Working with analog devices, Analog To Digital Converters (ADC) are used to translate continuous analog output signal into digital values. Affordable ADC devices have a finite resolution range of [8-16] bits and a sampling frequency of [100-800] Khz. For example an 10bit ADC can represent $2^{10} = 1024$ discrete values [2] that are spread over a *voltage range* (e.g, [1.0-3.0] V). This translates into 2mV per measurable quantity (e.g, mA). Those ADCs typically include [1-2] Least Significant Bit (LSB) error and suffer from integral and differential non-linearity issues (INL and DNL). In order to measure the energy consumed from time [T_1-T_2] (N measurements), the continuous integral should be replaced by a discrete Sum of the ADC samples multiplied by the sampling period T in this time range.

The ADC block is connected into a data acquisition device similar to a micro-controller, or a PC that features some sort of memory capacity. The polling of the sensor and the post-processing such as filtering, averaging and temporal logging takes place in this block. Depending on the type of the block connection, the end setup offers trade-offs, mainly in terms of cost, complexity and bandwidth. The next two subsections are dedicated to this analysis.

3.1 Open Loop Acquisition

The simplest way to perform ADC conversion is by placing an external converter (e.g, an oscilloscope) and observe the V_{Out} of the analog current sensor. The term *open-loop* is used because there is no feedback or any form of communication with the measured device itself. A precise example of this technique is shown in Figure 1. Among the advantages of this method is the reduced design effort, the potential large data point storage, the very high conversion bandwidth and the possibility of post-processing (e.g, filtering). In addition, the acquisition overhead (polling, filtering) is off-loaded into the external acquisition device, resulting into zero performance and power interference. On the other hand, the lack of feedback results into synchronization disability, since there is no information of the events that do happen in the target (measured) system. In addition the usage of the external device makes impossible to apply it in portable and battery critical applications. Therefore it is only valuable for device power characterization and platform evaluation.

3.2 Closed Loop Acquisition

General purpose System on Chips (SoC) typically include their own low cost ADC. Thus with the addition of a current sensor, it is possible for the system itself to measure the instant power that it currently dissipates. In this case, the acquisition device is a subset of the measured device itself.

As sampling frequency rises, polling the current sensor ADC becomes very expensive in terms of computational overhead. Even using asynchronous methods such as interrupts, an sampling frequency of 40khz can result in significant longer system response times for a 100Mhz simple RISC processor. The reader can argue that the specific sampling rate is an extreme example that exceeds the required information bandwidth. Since we use a non-ideal Operational Amplifier (Op-Amp), the amplified signal contains a lot of noise. A solution can be to use FFT and apply a low-pass filter to the signal. Since this requires expensive CPU computations we can use *oversampling*, a simple technique that collects more samples than the minimum sampling frequency and averages them periodically (in multiples of f_{adc}) to generate a smoother output [1].

4. EXPERIMENTAL SETUP

In this section we design, implement and evaluate a system based on a *Single Board Computer (SBC)*, capable of real-time self power and energy monitoring Figure 2. Our experimental setup solves the data acquisition and ADC problems discussed in Section 3 for Open and Closed loop acquisition approaches, by adding a simple external AVR microcontroller to handle the Analog to Digital Conversion and the data filtering of the analog current sensor. With this extension, the processing overhead that normally would be added into the main processor is off-loaded into the dedicated AVR board (DAB). In addition, the self-monitoring (closed loop) characteristic is maintained by adding a dedicated data communication channel and a set of signals to control the external AVR processor. The semantics of those commands are described in detail in Section 4.2. Effectively, our setup is a hybrid approach that balances between the advantages of closed and open data acquisition.

4.1 Implementation

In Section 2.1 we have pointed the degrade in power efficiency that an in-series resistor introduces when it is used to measure the power that a system dissipates. This is the reason why we have used a linear Hall Effect current sensor based on the *Allegro ACS712* that dissipates zero energy during operation. The ACS712 chip has a measurable range of [-5,+5] A. In addition a Texas Instruments OPA344 Op-Amp is featured to amplify the signal and create variable gain control. Table 1 summarizes the characteristics of the sensor board.

Table 1: ASC712 Hall Effect Sensor Characteristics

Parameter	Value
Primary Conductor Resistance	1mΩ
Rise Time	5μT
Frequency Bandwidth	80Khz
Total Output Error	1.5%

The dedicated Analog To Digital conversion and filtering board (AVR) is directly connected to the output of the analog sensor board (V_{Out}). It's primary operation principle is to gather converted values at a high rate, average them and periodically transmit them to the main processor. This particular running averaging procedure plays the role of the analog filter, by smoothing the output value [1]. In order for the main processor to be able to receive power values, we connect the dedicated AVR board with the ARM main board by UART serial interface. We have also added three signals between the AVR board and the main ARM processor to express the following operating modes:

- Operating Mode of the AVR co-processor [2 bit]
- Enable/Disable AVR co-processor [1 bit].
- Start/Stop Energy accounting session [1 bit].

The above signals are implemented using the General Purpose I/O (GPIO) interface that both controllers feature. The commands are issued from the main processor (ARM) to the co-processor (AVR). A detailed description of those commands exists in Section 4.2 in addition with the operational semantics. The analog to digital conversion rate of the co-processor is statically set at 40Khz with a conversion resolution of 12bit (4096 distinct values). The running average filter that is implemented features a range of 10 elements. This translates into an AVR to ARM data communication rate of 4Khz, when operating in continuous mode (See next section). From ARM side, both data and control can be easily accessed through the serial and GPIO conventional devices

4.2 Operational Semantics

The design decision to place the external co-processor for data conversion, temporal logging and filtering essentially transformed the AVR into a data server. We have designed the AVR firmware in a way that implements four operating modes for conversion and data logging allowing the main processor side designers to choose the method that better fits to their needs. The four operating modes are represented by the equivalent 2 bits of control signals, as shown in Figure 2.

- Single Power Shot Mode [milliwatts]: Perform a single conversion and transmit to the main processor the instant power consumption value. The conversion starts as soon as the start/stop signal is high.
- Continuous Power Mode [milliwatts]: Perform continuous conversion at a 4Khz and transmit the data to the main processor. The conversion starts and stops with the equivalent start/stop signal.
- Continuous Power Mode with Filter [milliwatts]: Perform continuous conversion at 40Khz, run a running average filter with 10 values range. For every averaged value transfer it to the main processor (4Khz rate)
- Energy Mode [millijoules]: Measure the Energy consumed from the rise until the fall of the start/stop signal and then transmit the result to the main processor. This function is implemented with a 32-bit

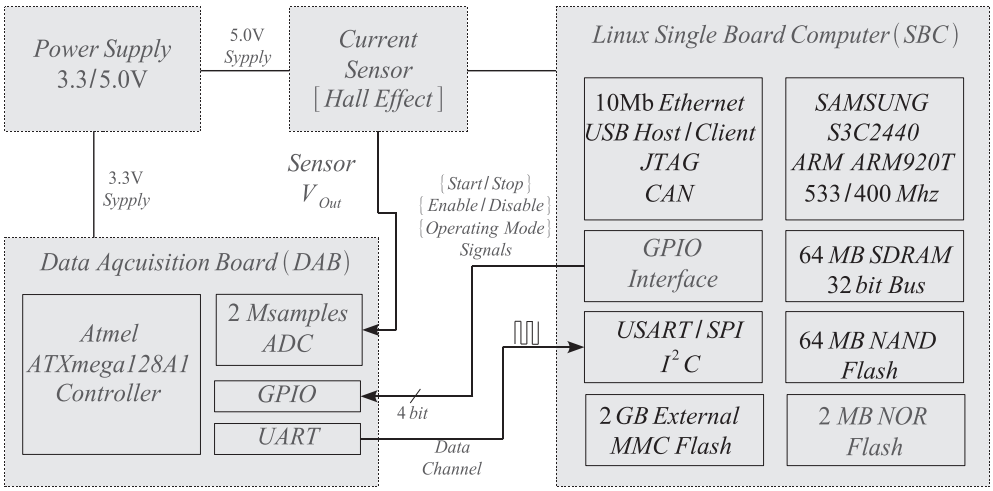

Figure 2: Block diagram of the self power monitoring setup with the dedicated power instrumentation Board.

wide variable yielding a very long wrap around time. The counter is only reseted when the mode changes.

5. EVALUATION

To validate our proposed power instrumentation setup Figure [2] we have performed a series of benchmarks on our main processor (ARM) board running a Linux kernel 2.6.13. Those benchmarks focus on the characterization of the following instrumentation parameters i) bandwidth, ii) response and iii) signal quality. In Table 2 we summarize our benchmarks (workload) and identify them as CPU, I/O intensive or hybrid.

Table 2: Performed Benchmarks

Workload	Type	Comments	T/S*
FFT	CPU	64k input	No
Bzip2	CPU + I/O	1MB Compress	No
NAND Flash Read	I/O	{1,2,3,4}MB	No
Web Browsing	CPU + I/O	Konqueror	Yes
ADD - NOP	CPU	Scheduler Test	No

*T/S = Touchscreen

On Figure 3 we compare two different workloads *Fast Fourier Transformation (FFT)* and *Bzip2 compress*. Those workloads have different behavior since FFT is purely CPU intensive in comparison to Bzip2 which is composed by a two phased execution i) read the whole input from external NAND flash and ii) perform the actual compression (write is done in main Memory). The CPU-bound nature of FFT is demonstrated into our measurements as we observe a stable high power consumption. The multi phase Bzip2 benchmark produces distinct power lobes that are lower compared to the FFT due to the stalls in the CPU.

In Figure 4 we demonstrate the power behavior of our test system when reading {1,2,3,4} MB of data from the NAND flash via FTP. In two cases (2 and 4 MB transfers) we observe an identical power lobe. This power activity is gen-

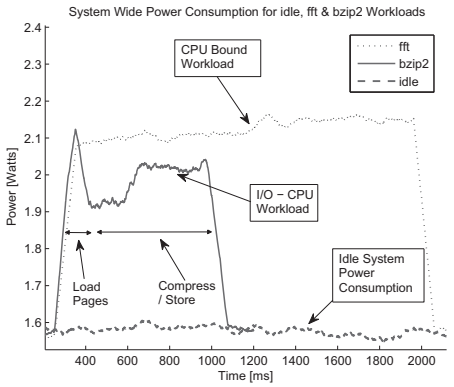

Figure 3: Power behavior of the FFT and Bzip2 compress benchmarks.

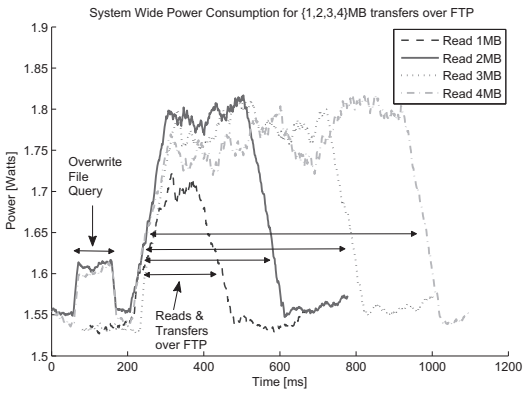

Figure 4: Reading from the NAND flash via FTP.

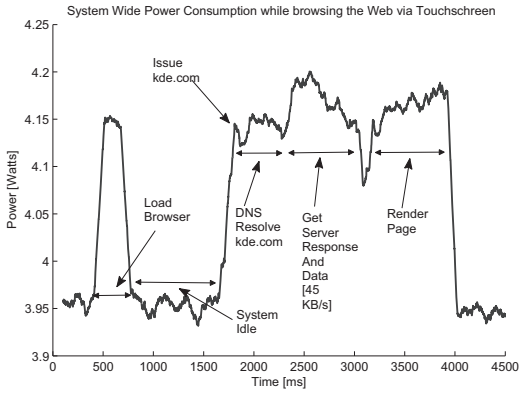

Figure 5: Browsing the web with a 7-inch LCD Touchscreen.

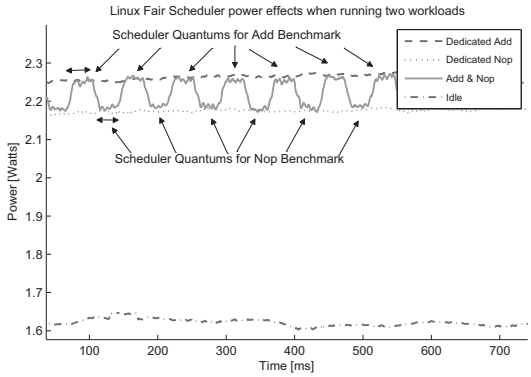

Figure 6: Fair Scheduler Impact on power consumption while running two processes concurrently.

erated by the overwrite file confirmation produced by our FTP client. Before every run, we reset our ARM board so we have a guarantee that the Linux buffer cache does not contain the requested data.

In Figure 5 we demonstrate a case study scenario, *web browsing*. This workload is performed into four phases i) launch the browser, ii) resolve symbolic address into IP address, iii) issue HTTP request and iv) get HTTP data and render into screen. In order our benchmark to be realistic, we have added an external 7-inch LCD Touchscreen with 100% back-light. Then we performed the four-step procedure we have described. The idle power consumption became almost *three times* greater compared to the system without the LCD Touchscreen. This is an important observation that points to a real hot-spot of portable electronic equipment.

In Figure 6 we demonstrate the power consumption of two hand-made application benchmarks designed to compare EPC's acquisition speed against the Linux scheduler time quantum. Those benchmarks are:

- A carefully written infinite loop that utilizes the CPU 100% by making exclusively arithmetic operations and Level 1 Cache hits.
- An infinite loop that utilizes the CPU 100% by performing exclusively No-Operation (NOP) instructions.

In the first phase of this experiment the above workloads run separate on our system without any interference. The power consumption of their dedicated runs was a straight line, significant higher for the arithmetic benchmark. In the second phase we have executed *both* of those workloads concurrently on the CPU. The default Linux scheduler assigned equal time quantums on both processes resulting in power consumption greater when the arithmetic benchmark was executing and lower when the NOP workload was running on the CPU.

6. CONCLUSIONS

In this work we have designed and implemented a real-time power measurement system based on an AVR controller and an analog Hall effect current sensor. The feature of a dedicated controller for analog to digital conversion and signal filtering enabled the implementation of four semantics for power and energy measurements. Those semantics make our proposed setup very efficient and easy to use, thus ideal for portable applications where power efficiency, system response and CPU performance are critical factors. In order to validate our proposed scheme we have used our system to instrument a *Linux Single Board Computer* based on an ARM microprocessor when running a selection of benchmarks. We have observed fast system response, quality output signal and very good accuracy characterizing our proposed setup as a well balanced measuring system between cost, and overall accuracy.

7. ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Union 7th Framework Programme [FP7/2007-2013] under the ICORES project (grant agreement no. 224759) In addition, we acknowledge Manolis Marazakis and Michael Ligerakis for their valuable assistance.

8. REFERENCES

[1] Atmel Corporation. Avr application note: Enhancing adc resolution by oversampling.

[2] Atmel Corporation. Avr1300: Using the atmel avr xmega adc.

[3] K. Bhatti, C. Belleudy, and M. Auguin. Power management in real time embedded systems through online and adaptive interplay of dpm and dvfs policies. In *Embedded and Ubiquitous Computing (EUC), IEEE/IFIP 8th Int. Conf. on*, pages 184 –191, dec. 2010.

[4] H. Forghani-zadeh and G. Rincon-Mora. Current-sensing techniques for dc-dc converters. In *Circuits and Systems, 2002. MWSCAS-2002. The 2002 45th Midwest Symposium on*, aug. 2002.

[5] V. Konstantakos, A. Chatzigeorgiou, S. Nikolaidis, and T. Laopoulos. Energy consumption estimation in embedded systems. In *IMTC 2006. Proceedings of the IEEE*, 2006.