IMTC 2006 – Instrumentation and Measurement
Technology Conference
Sorrento, Italy 24-27 April 2006

Energy consumption estimation in embedded systems

V. Konstantakos[1], A. Chatzigeorgiou[2], S. Nikolaidis[1], Th. Laopoulos[1]
[1]Electronics Lab. Physics Dept., Aristotle University of Thessaloniki, Thessaloniki, 54124, Greece,
[2]Dept. of Applied Informatics, University of Macedonia, Thessaloniki, 54006, Greece
Email: vkonstad@auth.gr

Abstract *– An energy consumption modeling technique for embedded systems based on a microcontroller is presented in this paper. The software tasks running on the embedded system are profiled and their characteristics are analyzed. The type of executed assembly instructions as well as the number of accesses to memory and to the A/D converter is the required information for the derivation of this model. An appropriate instrumentation set up has been developed for measuring and modeling the energy consumption in the corresponding digital circuits.*

Keywords *– power estimation, energy complexity, embedded systems, current measurement*

I. INTRODUCTION

The market of the embedded systems in microelectronics is ever increasing. Many applications are executed on platforms which incorporate embedded microprocessors, memory units and peripherals. Although performance still remains a basic design target, energy consumption has become a critical factor for such systems, especially after the explosion of the portable devices market. This situation has resulted to the requirement for accurate estimation of the energy consumed by the system for executing certain tasks. Although appropriate methods have been developed and certain models have been provided for the estimation of the energy consumption of embedded processors [1,2], there are not published up to now many efforts on modeling the energy consumption of a complete system and practically none of them is directly related to low power instrumentation applications.

In [1] the problem of modeling the energy consumption of a processor is examined for the first time. Appropriate energy budget is assigned to each instruction based on measuring the average current of the processor when it executes the specific instruction. Also the energy overhead (inter-instruction effect) resulted by the changing of the processor state as it executes the one instruction after the other is also modeled. The energy consumption for the execution of a software task results by summing the individual budgets of each executed instruction and the inter-instruction effects. Another approach in modeling the energy consumption in embedded pipelined processors based on measuring the instantaneous current of the processor has been developed in [2]. This approach found to present better accuracy and use an instrumentation set up [3] which can be applied to create models for other components, like memories and peripherals.

An energy consumption model for a system composed by a processor, an instruction and a data memory has been presented in [4]. This approach aims in defining the energy complexity of a program in a way analogous to the computational complexity. A polynomial expression of the number of executed assembly instructions (approximately mapping the number of primitive operations in the algorithm) and the number of memory accesses to the data and instruction memory is extracted by analyzing the program under study. Since the energy consumption of an independent assembly instruction can be measured and each access to a memory has a known energy cost, an estimate of the system's energy consumption is obtained as a single polynomial with appropriate coefficients.

A similar approach is used in [5], where the program whose energy consumption is to be analyzed, is represented by its unique control flow graph. Since the control flow graph of any structured program can be constructed by simply sequencing or nesting simpler graphs, an overall energy metric can be obtained by a hierarchical measure. By defining measures of executed instruction count and memory access count for primitive graphs and the operations of sequencing and nesting, a relatively simple software energy metric is extracted for any graph. Such a metric can be further used for comparing algorithms in terms of their energy consumption.

In [6] a complete methodology for estimating the energy consumption of a processor executing software tasks has been proposed. This methodology refers to in-order pipelined processors, which correspond to the most popular architecture used in embedded applications, and a model has been derived for the ARM7TDMI processor with accuracy up to 5%. This approach is based on instruction level energy models. The energy budget of each instruction as well as the energy overhead observed from executing one instruction after the other (this is due to the circuit state changes and is called inter-instruction effect) are estimated based on real measurements of the instantaneous current of the processor. To make the model reasonable appropriate grouping of the instructions has been applied so that instruction with similar characteristics to correspond to the same energy budget.

The proposed approach is differentiated in the sense that it targets a general purpose microcontroller usually employed for logging applications in low-power instrumentation systems. The microcontroller is connected to an A/D converter for input retrieval and to an external RAM for storing data. As such, the architecture as well as the operation of the microcontroller is expected to have a power consumption that is not analogous to that of typical

embedded processing units. Moreover the proposed model is not solely based on the number of executed instructions but rather on their number of cycles, the activity of the A/D converter and the number of memory accesses. Due to the nature of the microcontroller, energy consumption is estimated not simply by profiling the program to be executed but by parsing the program and analyzing the segments that initiate a read/write access to the memory.

II. SYSTEM ARCHITECTURE

The target platform (which is the one on which measurements have been performed) is shown in Figure 1. The general purpose microcontroller utilizes a data bus of 8 bits (the number of bits used for read/write is programmable) and an address space of 1024 x 1024 bytes is used. The control logic consists of six signals, three for accessing the memory and three for the A/D converter.

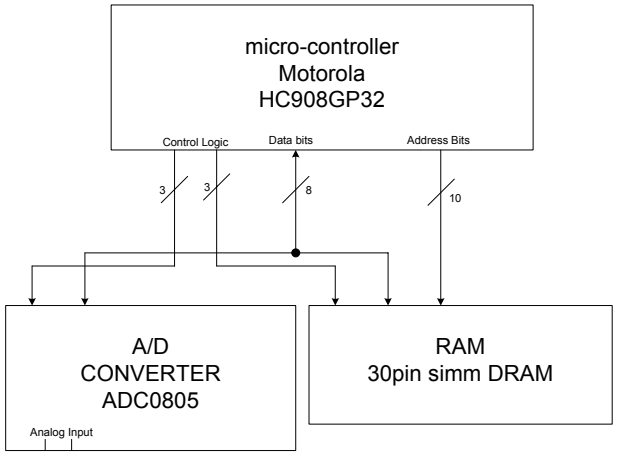

Fig. 1, Target Platform

This platform is a basic configuration of a wide range of instrumentation systems which are used in low-power applications. Similar components can be used in these applications, as for example D/A converters, input components like matrix keyboards and output components like 7-segment digits or LCD monitor screens. Generally, this is basic configuration of any system used for data logging applications. The characteristics of these applications are the non-intensive use of peripheral components. Usually scheduled measurements are conducted with use of the A/D converter. These measurements are processed by the microcontroller and at the end the results of the processing are stored to memory. So, nor the A/D converter or the memory is used intensively. Furthermore, memory module is not used during the processing period because the microcontroller has a small amount of internal memory that may be used for the duration of the calculations. The above configuration is quite simple but also quite indicative of such applications. The method that will be presented based on this system may be easily applied to similar systems.

III. THE PROPOSED APPROACH

In this work we propose an approach for modeling the power consumption of an embedded system and an instrumentation set up for the evaluation of its accuracy. According to the system components presented in the previous section, the total energy consumed by the system when it executes a software task can be expressed by the following equation:

$$E_{system} = E_{\mu Controller} + E_{RAM} + E_{ADC} \quad (1)$$

where $E_{\mu Controller}$ corresponds to the energy consumed by the microcontroller while executing a program, E_{RAM} corresponds to the energy consumed by the memory and E_{ADC} corresponds to the energy consumed by the A/D converter. Each component of the total energy will be analyzed below. The energy model for each system component has been based on separate physical measurements of the drawn current.

For the evaluation of the accuracy of the method, the measurement of the actual energy consumption is needed. Figure 2 shows the configuration of the system whose energy is to be measured during program execution. Every component has its' power supply pins connected to the measuring scheme, so that the drawn current is available for measuring.

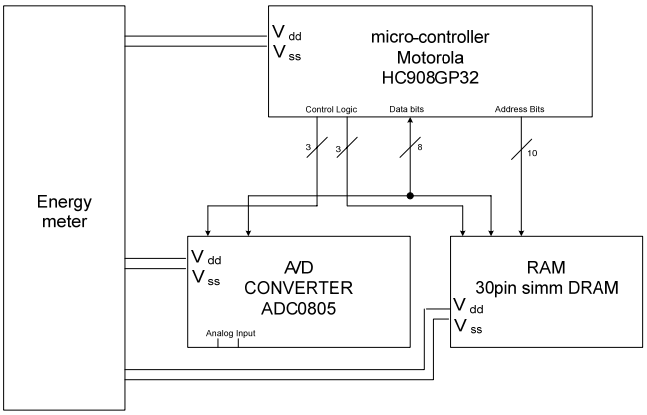

Fig. 2, System configuration

This is something which can not be done by using a simple ampere meter, as that used in some instruction level modeling approaches [1]. Besides, a measuring system is required that is capable of measuring accurately the energy consumed between certain clock cycles, which correspond to the beginning and the end of the execution of the software task. Such a circuit has been proposed in [7] and is illustrated in figure 3. This system is based on the integration of the current drawn by the system. The current of the processor as it is reflected by a current mirror is accumulated in a capacitor pair. When the one capacitor is full, it starts discharging and the current is directed in the second one which starts charging. In this way no current amount is lost. The capacitor fills are countered for a specific time window and the overall energy consumption can easily be estimated. Except of the

evaluation of the methodology accuracy, actual measurements are necessary to assign appropriate values to the coefficients that will be mentioned in the proposed model.

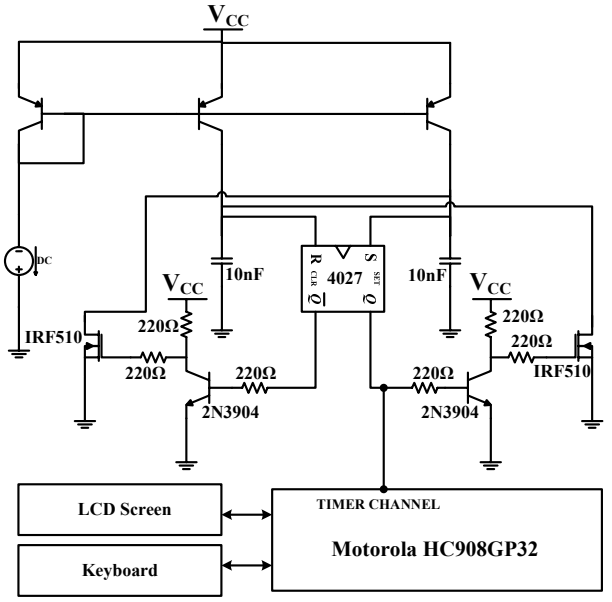

Fig. 3, Energy meter

Experimental measurements were conducted in order to investigate the power behaviour of the mentioned components. These measurements turned out to be quite similar for each specific component.

Concerning the energy consumption of the microcontroller, measurements have shown that energy does not depend that much on the instruction type but rather on the number of required instruction cycles. Typical measurement results are summarized in Table I. This table was created by measuring many different instructions and by taking into account different addressing modes and various arguments.

According to this observation, the proposed model is essentially a function of the number of executed instructions of each instruction group:

$$E_{\mu Controller} = f(\#instructions_{1\text{-}cycle}, \#instructions_{2\text{-}cycles}, \#instructions_{3\text{-}cycles}, \#instructions_{4\text{-}cycles}, \#instructions_{5\text{-}cycles}) = c_1.\#instructions_{1\text{-}cycle} + c_2.\#instructions_{2\text{-}cycles} + c_3.\#instructions_{3\text{-}cycles} + c_4.\#instructions_{4\text{-}cycles} + c_5.\#instructions_{5\text{-}cycle} + \quad (2)$$

where constants $c_1,\ldots,c_5$ correspond to the average energy for each instruction group.

Table I. Average Energy consumption of microcontroller instructions

Instruction Group	Avg. Energy (μJ)	Avg. Error
1 – cycle	0.0535	0.53%
2 – cycles	0.0566	2.85%
3 – cycles	0.0585	5.50%
4 – cycles	0.0580	3.19%
5 - cycles	0.0576	1.34%

The energy dissipation of the RAM memory consists in the active energy cost for each read/write access, the standby energy consumed due to the steady-state current flowing through memory cells and the energy dissipated during each memory row refresh. According to the measurements that have been performed a constant average value for each read and write access can be considered, without significant loss in accuracy. Each refresh operation can also be characterized by a standard energy cost. Concerning the steady-state current, its contribution to the overall energy should be calculated as the integral of power during the execution time of the program under study after subtracting the periods during which are read/write access is performed. However, for the sake of simplicity and because memory accesses in typical logging applications are rare compared to the total execution time, the proposed model integrates the standby power for the complete period of execution. Characterization measurement results are summarized in Table II. For the creation of this table, many different memory accesses were measured by taking into account various addresses and data to be exchanged with memory.

Table II. Average Energy consumption during memory operations and steady-state current value

Memory operation	Avg. Energy (μJ)	Avg. Error
read access	16.2	0.29%
write access	16.6	0.29%
refresh	3.40	
steady-state current: 8.43 mA		

The corresponding energy model can be formulated as:

$$E_{RAM} = f(\#read\_accesses, \#write\_accesses, \#refreshes + \int_{execution\ period} P_{standby} = c_1.\#read\_accesses + c_2.\#write\_accesses + c_3.\#refreshes + V_{dd} \cdot i_{steady\text{-}state} \cdot T_{clock\text{-}period} \cdot \#cycles \quad (3)$$

where c_1, c_2 and c_3 correspond to the average energy for read, write and refresh operations respectively, V_{dd} is the power supply voltage, T_{clock_period} is the selected clock period and #cycles the total number of cycles that correspond to the execution of the program under study. It should be mentioned, that in order to locate the segments of the code in which are read/write/refresh operation is performed, an appropriate parser should be used.

Finally, the energy consumption of the A/D converter consists also in an active component corresponding to the energy consumed for an access in order to retrieve data and a standby component corresponding to the current flowing through the converter when it is idle. Since the periods in which an access is made are a very small portion of the overall execution time (for the type of applications that have been considered) and because of the relatively limited time of an access (29μsecs), the energy due to the idle current is calculated by integrating the power for the total execution time. It should be noted, that the energy consumption for an access depends heavily on the transition that has to be made

on the A/D converter (from the previously sampled value to the current value). However, as already explained, due to the small number of accesses (compared to the total execution time of the microcontroller), a constant energy value has been considered as shown in Table III.

Table III. Average Energy consumption during A/D converter access and idle current

A/D conv. access	Avg. Energy (μJ)
read access	0.3
idle current: 0.912 mA	

The corresponding energy model is:

$$E_{ADC}=f(\#accesses)+\int_{execution\ period} P_{standby} =$$

$$c_1.\#accesses + V_{dd} . i_{idle} . T_{clock\text{-}period} . \#cycles \quad (4)$$

where c_1 corresponds to the average energy for an access to the A/D converter.

For deriving the energy consumption of a software task simply someone has to sum the above energy budgets for each system component which means that the application trace file is required to make estimations. However, the use of the trace file is not always convenient because of its large size and the need to compile and run the whole the application using a processor simulator. Instead, profiling methods (static and dynamic) should be used at the assembly code or even better at the higher language description level (e.g. C) of the application to extract appropriate information to guide the estimation procedure. Considering the models presented, such information corresponds to the number of executed instructions (for each instruction group), the number of memory accesses and refresh operations and the number of A/D converter accesses. The proposed methodology flow for the estimation of the energy consumed in a system executing a given software task is shown in Figure 4.

IV. CONCLUSIONS

In this paper an approach for modeling the energy consumption of embedded systems built around a microcontroller, a RAM memory and a A/D converter has been proposed. The software tasks which are executed by the microcontroller are analyzed by a profiling procedure and appropriate information about the activations of the system components and the used instructions is obtained. An appropriate instrumentation set up is employed for modeling the energy behavior of each component in the system both during its active state and during its standby operation. The total energy consumption is obtained by adding the individual energy amounts of these components. An energy meter circuit has been designed for the evaluation of the accuracy of the proposed approach. The measurements indicated that in such system the most important energy factor is the number of accesses to memory, while the contribution to the total energy consumption of the A/D converter and the microcontroller itself is less important. The proposed approach can be characterized as a useful tool for analyzing software routines in low-power applications and to a wide range of similar implementation systems.

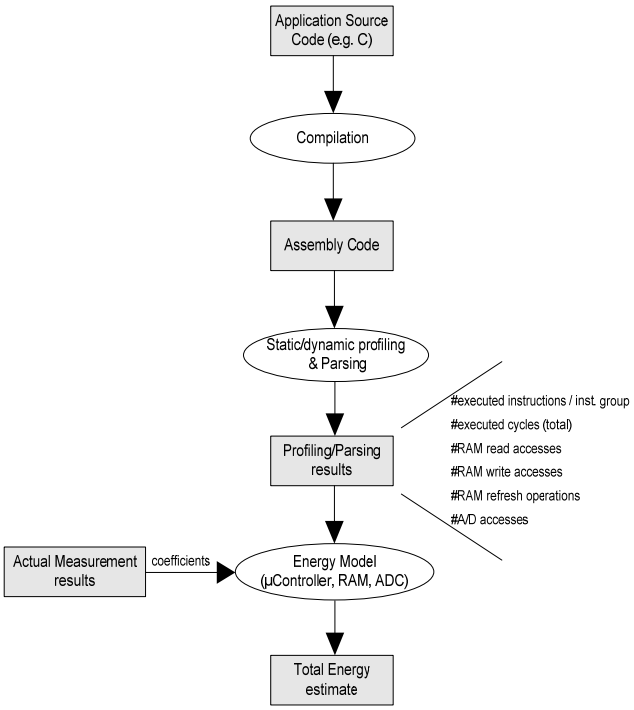

Fig. 4, Energy consumption estimation flow for embedded systems based on a microcontroller

REFERENCES

[1] Vivek Tiwari, Sharad Malik,and Andrew Wolfe, "Power Analysis of Embedded Software: A First Step Towards Software Power Minimization,"IEEE Transaction on Very Large Scale Integration (VLSI) Systems, Vol.2, No.4, pp. 437-445, December 1994

[2] N. Kavvadias, P. Neofotistos, S. Nikolaidis, K. Kosmatopoulos and Th. Laopoulos, " Measurement Analysis of the Software-Related Power Consumption in Microprocessors", IEEE Trans. on Instrumentation and Measurement, Vol.53.N.4, August 2004

[3] T. Laopoulos, P. Neofotistos, K. Kosmatopoulos, S. Nikolaidis, "Measurement of Current Variations for the Estimation of Software-related Power Consumption," IEEE Transactions on Instrumentation and Measurement, Vol 52, No 4, pp. 1206-1212, August 2003.

[4] K. Zotos, A. Litke, A. Chatzigeorgiou, S. Nikolaidis, G. Stephanides, "Energy Complexity of Software in Embedded Systems", IASTED International Conference on Automation, Control and Applications (ACIT-ACA 2005), Novosibirsk, Russia, June 20-24, 2005

[5] A. Chatzigeorgiou and G. Stephanides, "Energy Metric for Software Systems", Software Quality Journal, vol. 10, no. 4, December 2002, pp. 355-371

[6] S. Nikolaidis, N. Kavvadias, T. Laopoulos, L. Bisdounis and S. Blionas, "Instruction Level Energy Modeling for Pipelined Processors", Journal of Embedding Computing, Issue 3, Vol.1, 2005.

[7] V. Konstantakos, K. Kosmatopoulos, S. Nikolaidis, and Th. Laopoulos, "In-Chip Configuration for Monitoring Power Consumption in Micro-processing Systems", IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Sept. 2005