Run-Time Energy Estimation in System-On-a-Chip Designs*

J. Haid, G. Kaefer, Ch. Steger, R. Weiss
Institute for Technical Informatics
Graz University of Technology
Graz, Austria

Abstract - In this paper, a co-processor for run-time energy estimation in system-on-a-chip designs is proposed. The estimation process is done by using power macro-models, thus making analogue measurement equipment obsolete to the software engineer once the system-on-a-chip (SOC) design is characterized. Compared to sampling-based profiling systems [17], the performance overhead of energy profiling is less, because the energy estimation is done completely parallel to the functional units residing on the SOC. The proposed methodology can be used for run-time power optimization and in-system energy profiling. The co-processor was evaluated on a SOC for MPEG layer III audio decoding and the experimental results show a maximum relative error of <5%.

I. Introduction

Fast time to market and low energy consumption are major requirements for system-on-a-chip (SOC) designs developed for use in mobile applications as well as other embedded systems. The technological trends towards high-level integration and increasing performance, combined with the demand for energy-efficient system design, drive the development of energy estimation schemes.

In large VLSI circuits, such as System-On-a-Chip (SOC), it is often difficult to perform a run-time energy estimation of single functional units residing on the chip. Several approaches exist for simulating the energy consumption by using netlist simulators [9], instruction set simulators [4], and empirical methods [3]. In most cases these simulations rely either on an extensive pre-characterization effort or the need of hardware netlists, mostly not available to application engineers.

The goal of the presented work is to support the development of (i) energy-efficient software and (ii) power management policies without suffering from a significant amount of performance. This will lead to the design of more power-aware electronic systems meeting the tight constraints on energy consumption of battery-powered mobile systems, such as mobile phones and PDAs.

The major contributions of our work are: (i) development of a co-processor for energy profiling in embedded systems, (ii) run-time energy profiling with minimal impact on the overall performance. The *JouleDoc (JD) co-processor* performs an energy estimation of the entire SOC by energy accounting. In contrast to existing estimation schemes, the *JD co-processor* does the run-time energy estimation in parallel and makes any additional analogue measurement equipment obsolete after the one-time characterization process is done.

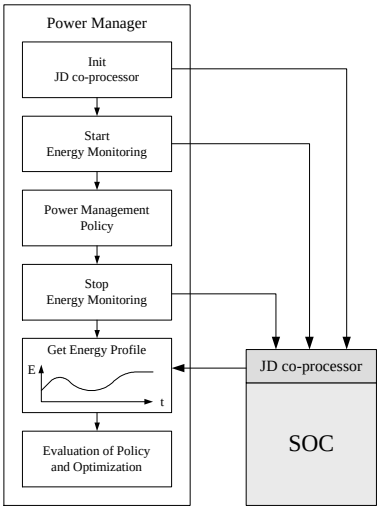

Fig. 1. A scenario using the *JD co-processor* for run-time optimization of the power management policy.

The energy estimation is done in parallel to the other components of the SOC, and a few instructions are required to perform energy profiling. The *JD co-processor* provides single instructions for starting and stopping the energy estimation of the entire SOC while a further single instruction performs energy estimation. Therefore the performance overhead of energy profiling is kept to a minimum.

Fig. 1 shows how a power manager is able to use the *JD co-processor* for optimization of its policy [1] by permanently tracking the impacts of the power manger's decisions on the SOC.

The remainder is organized as follows: in Section 2, related work is described. Section 3 presents the methodology and implementation of the *JD co-processor*. In Section 4 the evaluation of the *JD co-processor* is done by investigating an MPEG Layer III audio decoder. Section 5 concludes and discusses future work.

II. Related Work

While traditional profiling tools [17] gather program statistics in order to aid the software developer in optimizing the performance of the program, new tools are required to meet the needs of energy profiling systems [5] on electronic devices with limited energy resources. For effective

*This work was supported in part by austria*micro*systems AG, Austria.

profiling accurate and fast energy estimations are necessary. Various techniques based on instruction-level characterization and simulation of the underlying hardware have been proposed [8], [12]. Instruction-level power analysis, first proposed by Tiwari et. al. [6], relies on a base-cost model. The base-cost model is determined by running each instruction or short sequence of instructions in a loop and measuring the current/power consumption. Instruction-by-instruction energy costs are pre-characterized for each target processor. For energy estimation on instruction-level, instruction-set-simulators (ISS) are extended with energy models, as proposed in [4]. ISS have the disadvantages that they can execute only a limited number of instructions per second and often cannot be used for simulating complete SOC designs, because of simulation time. HW/SW co-simulators have been proposed [9] to reduce simulation time.

In [7] the authors present a run-time power estimation methodology which makes use of on-chip hardware counters. Hardware counters for tracking events are already integrated in commercial processors [18] primarily targeted for performance profiling [17]. The profiling process is based on system-wide sampling which is done by stopping the processor after a pre-specified amount of time or after a specified number of processor events. Our approach differs from [7] in three points: (i) the presented energy accounting hardware is not sampling-based and therefore does not interrupt the software running on the SOC, (ii) the co-processor estimates the energy consumption in hardware saving overhead in terms of performance and code size, (iii) the co-processor is designed for use in embedded systems with no high performance processors available.

III. JD Co-processor

A. *Power Macro-Modeling*

Modern SOC designs offer a various number of possibilities to reduce the energy consumption during run-time. Energy estimation is an important technique to evaluate the effectiveness of the implemented power aware features. As energy estimation of a SOC at gate level or even below is not practicable, abstraction layers of the system are introduced, such as the instruction-level of a processor. These abstraction levels allow the use of power macro-modeling as the fundamental theory for power estimation. Macro-modeling has already been used for RT-level hardware power analysis [10], [13], [16], [19]. Macro-modeling refers to the pre-characterization of a comprehensive set of gates, and the computation of its power dissipation and delay using either simulation or empirical methods. Power models for macro-blocks also utilize signal statistics at the boundaries of the macro-blocks, including bit-level statistics such as signal probabilities, transition probabilities, and spatial/temporal correlations [9].

For a set of macro-blocks M building a system S, the power consumption will be given by (1), where E_{SAVG} denotes the average energy consumption of S, and E_k the average energy consumption of the k-th macro-block as a function of its current state. Accurate E_k are determined by analysis of the macro-block, as done for micro-processors in [6], [7].

$$E_{SAVG} = \sum_k E_k(s_{M_k}) \qquad (1)$$

B. *Power Macro-Models for Power Estimation of SOC Designs*

A system-on-a-chip is a large VLSI circuit built of different analogue and digital components. For energy profiling the software designer is mostly interested in the energy consumption of each component on its own. For processor macro-modeling the instruction set is mostly used an abstraction layer, because ISS are available and can be extended with energy values.

While most energy estimation tools, like ISS, are implemented by people not involved in the hardware design process, the more effective way is to include the principles of macro-modeling in the hardware design flow. The component's designer mostly carries out numerous energy-related simulations at design time, leading to a more sophisticated definition of the energy-critical spots. While analogue simulation on component-level is possible, it cannot be done for large SOC in a reasonable time. Incorporating the energy related information at component-level leads to a bottom-up method contrary to other power estimation tools which "flatten" the design before analysis and re-structure the results after simulation has finished.

Furthermore, for the application engineer this granularity at component-level is comprehensible, and optimizations can be done by reducing the energy consumption component by component.

C. *Implementation*

The *JD co-processor* (Fig. 2) will be used for run-time energy analysis, profiling and monitoring of electronic systems. The co-processor is a sensor system with a control logic block to allow configuration and data transfer from/to the host controller. The primary design goal is to provide a tool with nearly no performance overhead.

C.1 *JD Energy Sensors*

The sensors within the *JD co-processor* are called *JD energy sensors*. Based on existing macro-models, the sensors count the occurrence of pre-specified operating conditions. Within the power macro-model different operating states (see section III.A) are associated to certain energy consumption values. Therefore, one can compare the *JD energy sensors* with small energy meters avoiding analogue circuits.

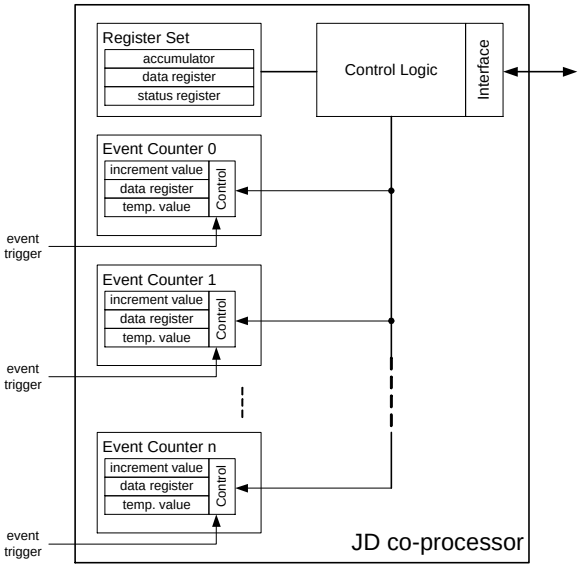

Fig. 2. Block diagram of the *JD co-processor.*

The average energy consumption E_{Mavg} of a macro-block is given by (2) where n_i denotes the number of occurrences of the operating condition S_i, E the energy consumption of the macro-block dependent on the operating condition s_i, and i is the number of different operating conditions for the macro-block M.

$$E_{Mavg} = \sum_i n_i \cdot E(s_i) \qquad (2)$$

C.2 JD Control Logic

The co-processor implements a small instruction set for data transfer and configuration tasks. Starting and stopping the energy estimation process is done by sending a single instruction to the co-processor. The *JD co-processor* is equipped with a general synchronous 32-bit interface. For specific host processors an interface wrapper must be implemented. Energy estimation of the whole SOC can be performed by a dedicated "read and accumulate" instruction, which adds the values of all counters in an accumulation register.

The program in pseudo-code (Fig. 3) shows the low overhead in lines of code that is required to perform run-time energy estimation. Except for the routine *configureCounters()* all commands are single 16-bit instructions. The calculation of the energy is done in parallel and requires 2*(number of integrated sensors) cycles; therefore, the host controller needs no additional program code for calculating the energy, compared to e.g. [7], and is able to run other tasks within this time.

The current implementation of a *JD energy sensor* requires 2100 gates and the control logic 3200 gates on an average 0.35µ process. The gate-count is dominated by the number of registers for storing adder values and accumulated data, which may be avoided by implementing an optimized memory system storing configuration and intermediate data.

```
configureJDProcessor(); //done once at startup
_startEnergyEstimation;

{
  … //program to be energy estimated
}
_stopEnergyEstimation;
_calculateEnergy;

//estimation is done in parallel and
//requires 2*(number of JD energy sensors) cycles

_readEnergyValue;
```

Fig. 3. Pseudocode for run-time estimation. All commands are single instructions except the configuration routine *configureJDProcessor*().

D. Energy Estimation Using FPGA-based Prototyping

Whole system evaluation is often done on prototype boards equipped with large FPGAs. Emulation with FPGAs is often used to provide nearly at-speed verification. However, emulation has several problems. A major problem is that FPGA's technology energy characteristics make it impossible to give accurate estimates of energy consumption for the eventual system fabricated in silicon. Other problems are long compile times, high costs, and performance. Nevertheless, FPGAs are gaining more and more importance in the design flow.

Once the designer decides to use FPGA technology for functional verification, the energy estimation can also be done functionally. In many cases the macro-models of the components can be built upon available information from data sheets, testchips, or simulations. Assuming the macro-models of the components exists, the designer must configure the related *JD energy sensors*, denoted as $E(s_i)$ in (2), and is able to perform energy analysis of the SOC in the prototyping phase without having the SOC available on silicon.

IV. Results

A. Power Macro-Models for a Multimedia SOC Design

For evaluation of the *JD co-processor* we use a SOC [21] that is thought to be an extension of portable devices like mobile phones, electronic organizers, or standalone battery-powered music players. The SOC (Fig. 4) consists of an RISC processor by ARC [20], an audio-subsystem, a memory system, and several interfaces. The processor is provided as a soft-macro in VHDL which offers us a wide range of possibilities for defining the processor's power-macro model. An audio subsystem is included on the chip as well as several interfaces to the PC, USB, SPI, and MultiMediaCard.

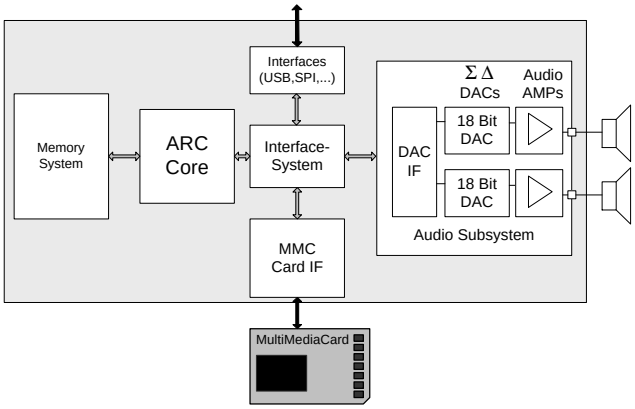

Fig. 4. Block diagram of the multimedia SOC for evaluation of the *JD co-processor[21]*.

For energy estimation we characterized the ARC processor including the different memories (XY, RAM, I-cache) at different available clock speeds of the on-chip PLL. Furthermore, the MultiMediaCard and Audio-Subsystem were described. The macro-model of the SOC consists of the following parameters: number of instruction fetches cycles, cache misses, store and load operations, pipelines stalls (all of ARC core), MMC load and store accesses, and sampling rate and volume of the audio-subsystem. The energy consumption is a function of the systems operating frequency f and the volume v of the audio subsystem.

$$E_S(f) = E_{CPU}+_{memories}(f) + E_{Audio}(f,v) + E_{MMC}(f) \quad (4)$$

The *JD co-processor* is mapped into the memory space of the processor by using the auxiliary register set (AUX registers), provided by the ARC core. The AUX registers are simple synchronous 32-bit registers designed for interfacing the ARC to further components.

B. *Energy Analysis of MP3 Audio Decoding Using an FPGA Prototyping System*

The digital part of the SOC including the *JD co-processor* is synthesized to a XILINX Virtex [14] FPGA while the analogue subsystem is available as a single testchip and connected to the FPGA (Fig. 5). The power values have been gathered by analysis of existing testchips.

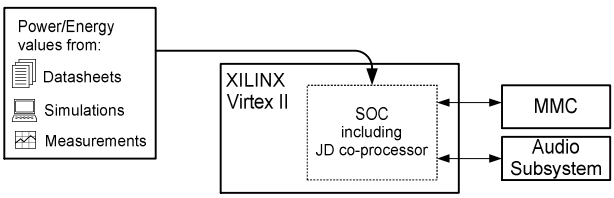

Fig. 5. Experimental setup. The FPGA is used for emulation of the digital part of the SOC including the *JD co-processor*.

The *JD co-processor* is evaluated by running the MPEG Layer III [15] audio decoder software on the SOC. The decoding algorithm consists of three blocks: frame unpacking, reconstruction, and inverse mapping. The first step is the synchronization and reading of the frame header followed by frame decoding. Requantization, stereo processing, if applicable, is done before applying the inverse modified cosine transformation (IMDCT) and the polyphase synthesis filterbank. The frames are loaded from the MMC via a FIFO. The PCM audio samples are also written onto the audio subsystem using a FIFO.

The experiments are done on audio files with 48kHz, 44kHz, and 32kHz sampling rate, using a bitrate of 128 kBits/sec. Fig. 6 shows the energy consumption of the SOC while playing 1 second of a MPEG Layer III files.

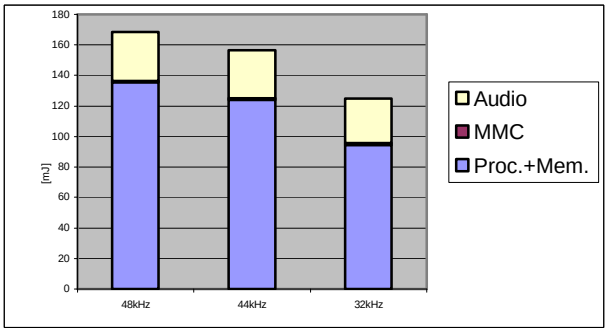

Fig. 6. Estimated energy consumption for playing 1 second of MPEG Layer III file.

Table 1 shows the results of the energy estimation for each subroutine. Therefore the commands for starting and stopping the *JD energy sensors* are placed nearby the subroutine calls. As these commands consist of only two assembler-code lines the influence on the algorithm is minimal. The MMC's and audio subsystem's energy consumption is not taken into account for this analysis, because they do not directly influence the subroutines.

TABLE 1
ENERGY ESTIMATIONS OF THE ARC CORE BY THE *JD CO-PROCESSOR* FOR SINGLE SUBROUTINES OF THE MP3 AUDIO DECODER.

	48kHz 128 kBit/s [mJ]	44kHz 128kBit/s [mJ]	32kHz 128kBit/s [mJ]
GetScaleFactors	0,0141	0,0124	0,0139
Stereo	0,0250	0,0229	0,0242
Reorder	0,0366	0,0320	0,0335
Antialias	0,0604	0,0551	0,0577
HuffmanDecode	0,1025	0,0970	0,1023
DequantizeSample	0,1619	0,1478	0,1561
Hybrid	0,3723	0,3446	0,3588
SubBandSynthesis	0,8708	0,8029	0,8470

TABLE 2
RELATIVE ERROR OF ESTIMATED ENERGY CONSUMPTION OF *JD CO-PROCESSOR* COMPARED TO TESTCHIP MEASUREMENTS.

	48kHz 128 kBit/s [%]	44kHz 128kBit/s [%]	32kHz 128kBit/s [%]
GetScaleFactor	-1,63	-0,31	-0,15
Stereo	1,09	2,47	3,77
Reorder	1,16	0,95	2,44
Antialias	-0,22	1,96	2,10
HuffmanDecode	-4,84	-3,92	-3,67
DequantizeSample	1,79	2,84	4,52
Hybrid	-3,62	-4,55	-2,68
SubBandSynthesis	2,28	4,33	3,52

The accuracy of the energy estimations of the JD is determined by analyzing several subroutines of the MP3 decoder. Table 2 shows the relative error of the estimated energy consumption compared to empirical measurements done with a multimeter (running the subroutines in loops).

The relative error of the estimation results is within a range of ±5% compared to the actual energy consumption. In our opinion this is accurate enough for the evaluation of power management policy strategies during run-time and source code optimizations at the software development phase.

An additional point of interest involves the overhead of the co-processor in terms of gate-count resp. area. Therefore, the co-processor has been synthesized using an available 0,35µm process [21]. The *JD co-processor* equipped with 8 *JD energy sensors* requires about 12000 gates which results in an area of approximately $1mm^2$ on the chip. A reduction of the area can be obtained by using an optimized register file instead of unstructured sequential logic.

V. Conclusion

Time-to-market constraints and the increasing number of SOC-based mobile electronic devices require new methodologies for energy estimation. In this paper we have presented the implementation of the *JD co-processor* delivering run-time energy estimations using the macro-modeling approach. The *JD co-processor* can be used for supporting the power manager's decisions during run-time and offers the possibility to continuously monitor the energy consumption without losing a significant amount of performance. The experiments with a SOC for audio decoding have shown a maximum relative error of 5% which makes the appliance and evaluation of power-aware strategies [2] practicable.

Future work will include both implementing power managers that exploit the permanent information of the energy consumption for optimizing the policy as well as investigating possibilities to reduce the size of the *JD co-processor*.

VI. References

[1] L. Benini, A. Bogliolo, G.A. Paleologo and G. De Micheli, "Policy optimization for dynamic power management," *IEEE Trans. on Computer-Aided Design*, Vol. 18, No. 6 (1999), pages 813-833.

[2] L. Benini and G. de Micheli, "System-Level power optimization: Techniques and Tools," *ACM Transactions on Design Automation of Electric Systems*, Vol. 5, No. 2, April 2000.

[3] J. Flinn and M. Satyanarayanan, "PowerScope:A Tool for Profiling the Energy Usage of Mobile Applications," *In Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pp.23–30, 1999.

[4] T. Simunic, L. Benini, G. De Micheli, "Cycle-Accurate Simulation of Energy Consumption in Embedded Systems," *In Proc. Design Automation Conference*, pp.867–872, 1999.

[5] T. Simunic, " Energy Efficient System Design and Utilization," PhD-Thesis, Stanford University, 2001

[6] V. Tiwari, S. Malik, A. Wolfe, "Power Analysis of Embedded Software: A First Step Towards Software Power Minimization," *IEEE Transactions on VLSI Systems*, vol. 2, no.4, pp.437–445, December 1994.

[7] Russ J. and M. Martonosi, „Run-Time Power Estimation in High Performance Microprocessor," *In Proc. International Symposium on Low Power Electronics and Design*, pp. 135-140, August 2001.

[8] T. Sato, Y. Ootaguro, M. Nagamatsu, and H. Tago, "Evaluation of architecture-level power estimation for CMOS RISC processors," *In Proc. Symp. Low Power Electronics*, pages 44–45, Oct. 1995.

[9] M. Lajolo, et.al., "Efficient Power Estimation Techniques for HW/SW Systems," *In Proc. Proc. Design Automation and Test Europe (DATE)*, March 2000.

[10] J. Rabaey and M. Pedram (Editors). Low Power Design Methodologies. Kluwer Academic Publishers, Norwell, MA, 1996.

[11] Sinha A., et.al., "JouleTrack - A Web Based Tool for Software Energy Profiling, "*In Proc. Design Automation Conference*, Las Vegas, Nevada, USA, June 2001.

[12] C-T Hsieh, L-S. Chen, and M. Pedram, "Microprocessor power analysis by labeled simulation," *Proc. of Design Automation and Test in Europe*, Mar. 2001, pp. 182-189.

[13] Q. Qiu, Q. Wu, and M. Pedram, "Cycle-Accurate MacroModels for RT-Level Power Analyis," *Proceedings of 1997 International Symposium on Low Power Electronics and Design*, pp. 125-130, August 1997.

[14] XILINX Inc. http://www.xilinx.com.

[15] ISO/IEC JTC 1/SC 29/WG 11 11172-3. Information Technology – Coding of moving pictures and associated audio for digital storage media up to 1.5 Mbit/s – Part 3: Audio. International Organization for Standardization, November 1994.

[16] Q. Wu, C. Ding, C. Hsieh, and M. Pedram, "Statistical Design of Macro-models For RT-Level Power Evaluation," *In Proc. of the Asia and South Pacific Design Automation Conference*, pp.523-528, January 1997.

[17] VTUNE Performance Analyzer. Intel Corporation, 2002.

[18] The IA-32 Intel® Architecture Software Developer's Manual, Volume 3: System Programming Guide. Intel Corporation, 2002.

[19] L. Benini, D. Bruni, M. Chinosi, C. Silvano, V. Zaccaria, R. Zafalon, "A Power Modeling and Estimation Framework for VLIW-based Embedded Systems," *In Proc. Int. Workshop on Power And Timing Modeling, Optimization and Simulation PATMOS*, September. 2001.

[20] ARC International. *http://www.arccores.com*

[21] austriamicrosystems AG. *http://www.austriamicrosystems.com*