Cycle-Accurate Simulation of Energy Consumption in Embedded Systems

Tajana Šimunić, Luca Benini* and Giovanni De Micheli
Computer Systems Lab, Stanford University
*DEIS University of Bologna, Italy
tajana@polaris.stanford.edu

Abstract

This paper presents a methodology for cycle-accurate simulation of energy dissipation in embedded systems. The ARM Ltd. [1] instruction-level cycle-accurate simulator is extended with energy models for the processor, the L2 cache, the memory, the interconnect and the DC-DC converter. A SmartBadge, which can be seen as an embedded system consisting of StrongARM-1100 processor, memory and the DC-DC converter, is used to evaluate the methodology with the Dhrystone benchmark. We compared performance and energy computed by our simulator with measurements in hardware and found them in agreement within a 5% tolerance. The simulation methodology was applied to design exploration for enhancing a SmartBadge with real-time MPEG feature.

1 Introduction

Energy consumption and power dissipation are critical factors in embedded system design. Peak power dissipation sets constraints on thermal and power supply design for the system. Average power consumption is directly related to battery life, hence it may be the critical factor that sets system weight and cost. CAD tool support is needed to evaluate performance and energy consumption in portable embedded system designs.

Most CAD work to date has focused on energy estimation of *systems on a chip* (SOC) or component-level power reduction. On the other hand, many portable embedded systems are built from commodity components. Such systems have to rely on the data sheets to estimate both performance and energy consumption instead of the detailed capacitance models assumed by the designers of SOCs or ASICs. As a result, most of the tools developed for the SOC or ASIC community are not applicable. This paper describes a methodology for performance and energy consumption simulation of embedded systems based on discrete components for which only simple black-box power models are available.

The primary motivation for this work comes from our experience with the redesign of a SmartBadge [2]. As far as the research work described in this paper is concerned, the detailed technical specifications of the SmartBadge are inessential. It suffices to view the SmartBadge with the system model consisting of a microprocessor with level-1 (L1) cache, level-2 (L2) cache, off-chip memory and DC-DC converter connected with the interconnct, which also represents the basic configuration of most portable devices. A prototype implementation of the SmartBadge on a PCB was available to us. The design task we were confronted with was to enhance the prototype implementation by adding other components such as real-time MPEG video decode. Since the original hardware did not meet either the performance or the energy consumption constraints when running the MPEG decode algorithm, we had to look into ways to redesign both the hardware and the software architectures, while keeping the energy consumption under tight control.

Design process for such a portable embedded system starts with the selection of the commodity components that may meet the performance and the energy consumption criteria based on the data sheets. Typicaly only a few processor families can be evaluated due to resource and time limitations. In addition, many companies often license an architecture and as a result prefer to focus designs on the processor family licensed. We selected the ARM processor family [1] to illustrate the methodology for cycle-accurate energy consumption simulation used in the design of our portable device.

Cycle-accurate instruction-level simulators are used for performance estimation of the software portion of the design industry-wide. Whole system evaluation is often done on prototype boards. Due to long design times and costs for prototype board design, only a few hardware architectures can be tried. We needed to use the simulation-based methodology in order to easily explore many different hardware and software architectures and get accurate performance and energy consumption estimates. As a result, we extended the basic instruction-level simulator provided in the ARM software development kit with the cycle-accurate energy consumption models for the processor with the level 1 and the level 2 caches, the off-chip memories, the interconnect and the DC-DC converter. The methodology presented in this paper can be applied to any cycle-accurate instruction-level simulator.

A brief discussion of the existing embedded system modeling approaches is given in Section 2. System model and the methodology for cycle-accurate simulation of energy dissipation are presented in Section 3. Section 4 shows that the simulation results of timing and energy dissipation using the methodology presented are within 5% of the hard-

DAC 99, New Orleans, Louisiana

ware measurements for the Dhrystone test case. Hardware and software architecture trade-offs for SmartBadge's real-time MPEG video decode design are explored using cycle-accurate energy simulation in Section 5.

2 Related Research

Embedded system design is an emerging area with traditionally limited CAD support for performance and energy consumption estimation. Recently, several commercial tools have been announced in this area, focusing on the integration of the system components into one simulation engine and on the generation of the communication interface between the components [3, 4, 5, 6]. These tools target mainly functional verification and, to some extent, performance estimation.

Extensive research work has been done on power modeling at the architecture level. For instance, Landman et al. [7] presented activity-sensitive power analysis at the netlist level for the data path, memory, the control path and the interconnect. Liu [8] showed architecture-level power analysis for systems on a chip. Even though these approaches and several similar others can be very helpful for chip designers, we are tackling the power estimation problem at a higher abstraction level. In our model, system components are chips or sets of chips available on the market and not custom-designed functional blocks.

In addition to architecture energy consumption models, there has been research in energy and performance modeling for memories and microprocessors. Accurate energy consumption cache models are presented by Kamble et al. [9]. The cache models rely on knowledge of capacitances of each portion of the cache design, stochastical distributions for signal values and the run time statistics for hits/miss and read/write counts. Wilton and Jouppi [10] designed CACTI - an enhanced cache access and cycle time model based on resistance and capacitance values derived from the technology files and the cache netlist. RAM energy consumption and performance models based on technology parameters and the netlists are described by Itoh et al. [11]. These models are not applicable in our framework because they would require knowledge of the internal structure and implementation of commodity components.

Instruction level power analysis presented in [12] by Tiwari et al. provides a way of estimating the energy consumption of software by measuring the energy consumption of each instruction when running on the processor of interest. Non-ideal instruction execution (eg. pipeline stalls) is modeled by measuring the additional current consumption caused by the combination of instructions that cause the event of interest to occur. Wan [13] extended StrongARM processor model with base current costs for each instruction. The average power consumption for most of the instructions is 200mW measured at 170MHz. Load and store instructions required 260mW each. Non-ideal effects, such as stalls due to register dependencies, and cache effects were not considered by Wan. If all effects were measured, the total power consumed per instruction would match the data sheets. Because the difference in energy per instruction is minimal, it can be expected that the average power consumption value from the data sheets is on the same level of accuracy as the instruction-level model.

The system-level energy simulator proposed in [14] models every resource as a state machine where each state represents specified performance and power consumption. The total system power dissipation is obtained by summing the contribution of each component. The main limitation of this approach is that it completely abstracts functionality away, hence it has limited accuracy.

There have been a few tools [15, 16], that estimate the energy consumption of software, caches and off-chip memory in SOC design. Performance and energy consumption of each component are separately analyzed. The final system energy consumption is obtained by summing the results of each analysis. Energy consumption models used by these approaches require detailed knowledge of the internal structure and implementation of the components and as such are not applicable to designs based on commodity parts. In addition, it is difficult to estimate their accuracy since no comparison was given of the simulation results with the hardware measurements.

The methodology presented in this paper combines performance and energy estimation models of each system component into one cycle-accurate instruction-level simulator. The whole system is simulated as a unit, providing cycle-by-cycle timing and energy consumption values of each component. Since component interaction directly affects both performance and energy consumption, the integrated approach presented in this paper is more accurate in its estimates than the previous approaches. Power models of the individual components have been completely inferred from the data-sheet information. Nevertheless, our estimates are accurate: the results presented in Section 4 show agreement with measurements within 5% of accuracy.

3 System model

The SmartBadge considered in this paper can be modeled as a typical embedded system consisting of a microprocessor with two levels of cache, off-chip memory and DC-DC converter connected with the interconnect. Various types of memories are modeled - SRAM, DRAM, burst SRAM and DRAM, FLASH, with or without customizable L2 cache. Selection of the best hardware architecture and software organization given energy and performance constraints is done with help of an instruction-level simulator that has been extended with the energy models for all system components. We estimate not only average power, but also cycle-by-cycle energy.

Since in our design we have selected the ARM processor family, we implemented the energy models as extensions to the cycle-accurate instruction-level simulator for the ARM processor family, called the ARMmulator [1]. Figure 1 shows the simulator architecture. The ARMulator is normally used for functional and performance validation. The typical sequence of steps needed to set up system simulation can be summarized as follows. (i) The designer provides a simple functional model for each system component other than the processor (ii) The functional model is annotated with a cycle-accurate performance model (iii) Application software (written in C) is cross-compiled and loaded in specified locations of the system memory model. (iii) The simulator runs the code and the designer can analyze execution using a cross-debugger or collecting statistics. A designer interested in using our methodology would only need to additionally provide cycle-accurate energy models for each component during step (ii) of the simulation setup. Thus, the designer can obtain power estimates with little incremetal effort.

On each cycle of execution the ARMulator sends out the information about the state of the processor ("cycle type") and its address and data busses. Two main classes of processor cycle types are *processor active*, where active power

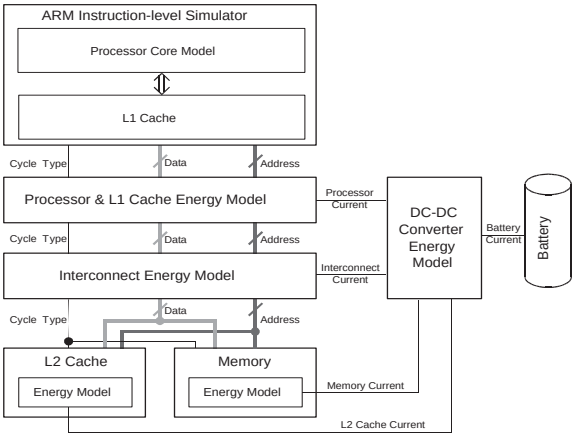

Figure 1: Simulator Architecture

is consumed, and *processor idle*, where idle power is consumed. The idle state in our model represents either an off-chip memory request or an ARMulator debugger request. The off-chip memory request first passes through the interconnect energy model where energy consumed by the interconnect and pins is calculated based on the number of lines switched during the cycle on the data and address busses. Energy models of both L2 cache and memory account for active and idle states. L2 cache, when present, is always accessed before the main memory and so is active on every memory access request. L2 cache and memory model introduce processor idle cycles corresponding to their access times. The DC-DC converter energy model sums all the currents consumed each cycle by other system components, accounts for its efficiency loss, and gets the total energy consumed from the battery. The total energy consumed per cycle is the sum of the component energy consumptions:

$$E_{Cycle} = E_{Proc.} + E_{Interc.} + E_{Mem.} + E_{DC/DC} + E_{L2\ Cache} \quad (1)$$

The total energy consumed during the execution of the software on a given hardware architecture is the sum of the energies consumed during the each cycle. Models for energy consumption and performance estimation of each system component are described in the following sections.

3.1 Processor

The ARM simulator provides a cycle-accurate, instruction-level model for each ARM processor and L1 on-chip cache. The model was enhanced with energy consumption estimates based on the information provided by the data sheets. Two power states are considered: active state in which processor is running with the on-chip cache, and the state in which the processor is executing NOPs while waiting to fill the cache.

When the processor is executing with the on-chip cache, it consumes the active power specified in the data sheet P_m measured at given voltage V_m and frequency of operation f_m. Total active capacitance within the processor, $C_{proc,a}$, is estimated as:

$$C_{proc,a} = \frac{P_m}{V_m^2 f_m} \quad (2)$$

The amount of energy consumed by processor and L1-cache at specified processor cycle time T_{cycle} and CPU core voltage V_{cc} is:

$$E_{Processor,active} = P_{proc,a} T_{cycle} = C_{proc,a} V_{cc}^2 \quad (3)$$

When there is an on-chip cache miss, the processor stalls and executes NOP instructions which consume less power. $C_{proc,NOP}$ can be estimated from the power consumed during execution of NOPs $P_{proc,NOP}$ at voltage V_m and frequency f_m:

$$C_{proc,NOP} = \frac{P_{proc,NOP}}{V_m^2 f_m} \quad (4)$$

The energy consumed within processor core per cycle while executing NOPs is:

$$E_{Processor,NOP} = C_{proc,NOP} V_{cc}^2 \quad (5)$$

3.2 Memory and L2 cache

The processor issues an off-chip memory access when there is a L1 cache miss. The cache-fill request will either be serviced by the L2 cache if one is present in the design or directly from the main memory. On L2 cache miss, a request is issued to the processor to fetch data from the main memory. The time and energy penalty accrued are from both the access to L2 cache and main memory access. Data sheets specify the memory and L2 cache access times, and energy consumed during active and idle states of operation.

Memory access time, T_{mem}, is scaled by the processor cycle time, T_{cycle}, to obtain the number of cycles the processor has to wait to serve a request, N_{wait} (Equation 6). Wait cycles are defined for two different types of memory accesses: sequential and non-sequential. Sequential access is at the address immediately following the address of the previous access. In burst type memory the sequential access is normally a fraction of the first, non-sequential, access.

$$N_{wait} = \frac{T_{mem}}{T_{cycle}} \quad (6)$$

Two energy consumption states are defined for each type of memory: active and idle. Energy consumed per cycle while memory is in active state operating at supply voltage V_{dd} is:

$$E_{Memory,active} = \frac{C_{mem} V_{dd}^2}{N_{wait} + 1} \quad (7)$$

Active memory capacitance, C_{mem}, can be estimated from the active power specified in the data sheet, P_{mem}, measured at voltage V_m and frequency f_m:

$$C_{mem} = \frac{P_{mem}}{V_m^2 f_m} \quad (8)$$

Idle state can be further subdivided into multiple states that describe modes of operation for different types of memories. For example, DRAM might have two idle states: refresh and sleep. The designer specifies the percentage of the time ρ_i memory spends in each idle state. Total idle energy per cycle for memory is:

$$E_{Memory,idle} = T_{cycle} \sum_{i=0}^{n} P_i \rho_i \quad (9)$$

where P_i is power consumption in idle state i.

The L2 cache access time and energy consumption are treated the same way as any other memory. L2 cache organization is determined from the number of banks, lines per

bank, and words per line. Line replacement is random, the cache is integrated data and instruction with write-through policy. Cache hit rate is strongly dependent on its organization, which in turn affects the total memory access time and the energy consumption.

3.3 Interconnect and Pins

The interconnects on the PCB can contribute a large portion of the off-chip capacitance. Capacitance per unit length of the interconnect is a parameter in the energy model that can be obtained from the PCB manufacturer. The length of an interconnect can be estimated by the designer based on the approximate placement of the selected components on the PCB. Pin capacitance values are reported on the data sheets.

The hardware prototype of the SmartBadge uses a standard PCB [17] with line delay of $71ps/cm$ and stripline and microstrip capacitances of 1.6 and $1.1pF/cm$ respectively. The interconnect cross-sections have the following parameters: $h = 0.2mm$ for microstrip and $0.4mm$ for stripline, $w = 0.24mm$, $t = 0.018mm$, and $\epsilon_r = 4.3$. Propagation delay can usually be neglected relative to the memory response time. On the other hand, the total interconnect capacitance needs to be considered since even at smaller interconnect lengths it is comparable to the capacitance of an I/O pin.

For each component the average length of the clock line, data and address buses between the processor and the component are provided as one of the input simulation parameters. Hence, the designer is free to use any wire-length estimate [18] or measurement. The interconnect lengths used in our simulation of SmartBadge come from the prototype board layout.

The total capacitance switched during one cycle is shown in Equation 10. It depends on the capacitance of one interconnect line and the pins attached to it, C_{switch}, and the number of lines switched during the cycle, N_{switch}.

$$C_{line} = N_{switch} C_{switch} \tag{10}$$

The total energy consumed per cycle, $E_{Interconnect}$, is a function of the voltage swing on the lines that switched, V_{dd}, total capacitance switched, C_{line}, and the total time to access the memory, $N_{wait} + 1$:

$$E_{Interconnect} = \frac{C_{line} V_{dd}^2}{N_{wait} + 1} \tag{11}$$

3.4 DC-DC Converter

DC-DC converter losses can account for a significant fraction of the total energy consumption. Figure 2 shows the dependence of efficiency on the DC-DC converter output current. Total current drawn from the DC-DC converter by the system each cycle, I_{out}, is a sum of the currents drawn by each system component. A component current, I_c, is defined by:

$$I_c = \frac{E_c}{V_c T_{cycle}} \tag{12}$$

where E_c is the energy consumed by the component during cycle of length T_{cycle} at operating voltage V_c.

Total current drawn from the battery, I_{bat} can be calculated as:

$$I_{bat} = \frac{I_{out}}{Eff} \tag{13}$$

Efficiency, Eff, can be estimated using linear interpolation from the data points derived from the output current versus efficiency plot in the data sheet. Total energy consumed by the battery each cycle is:

$$E_{Cycle} = I_{bat} V_{bat} T_{cycle} \tag{14}$$

The energy consumed by the DC-DC converter, $E_{DC/DC}$, is difference between the energy provided by the battery, E_{Cycle} and the energy consumed by all other components, E_{out}:

$$E_{DC/DC} = E_{Cycle} - E_{out} \tag{15}$$

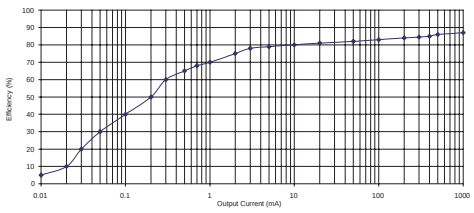

Figure 2: DC-DC Converter Efficiency

4 Validation of the Simulation Methodology

We validated the cycle-accurate power simulator by comparing the computed energy consumption with measurements on the SmartBadge prototype implementation. The SmartBadge prototype consists of the StrongARM-1100 processor, DC-DC Converter, FLASH and SRAM on a PCB board. All the components except the CPU core are powered through the 3.3V supply line. CPU core runs on 1.5V supply. DC-DC converter is powered by the 3.5V supply. DC-DC converter efficiency table contains 22 points derived from the plot shown in Figure 2. Stripline interconnect model is used with $1.6pF/cm$ capacitance. Table 1 shows other system components. Average current consumed by the processor's power supply and the total current drawn from the battery are measured with digital multimeters. Execution time is measured using the processor timer.

Table 1: Dhrystone Test Case System Design

Component Units	Cycle T. (ns)	Active P (mW)	Idle P (mW)	Pin Cap. (pF)	Line L. (cm)
SA-1100	5-20	400	170	5	N/A
FLASH (1MB)	80	74	0.5	10	2
SRAM (1MB)	90	55	0.01	8	3

Industry standard Dhrystone benchmark is used as a vehicle for methodology verification. Measurements and simulations have been done for ten different operating frequencies of SA-1100 and SA-110 processors. Dhrystone test case is run 10 million times, 445 instructions per loop. Simulations ran on HP Vectra PC with Pentium II MMX 300 MHz processor and 128 MB of memory. Hardware ran 450 times faster than the simulations without the energy models. Simulations with energy models were slightly slower. Figure 3 show average processor core and battery currents. Average simulation current is obtained by dividing the total energy consumed by the processor core or the battery with their respective supply voltages and the total execution time.

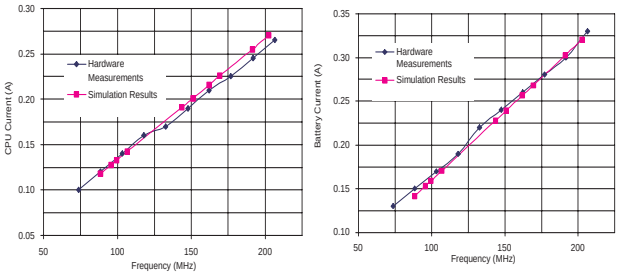

Figure 3: Average Processor Core and Battery Currents

Simulation results are within 5% of the hardware measurements for the same frequency of operation. The methodology presented in this paper for cycle-accurate energy consumption simulation is very accurate and thus can be used for hardware and software architecture design exploration in embedded system designs. An example of such exploration is presented next.

5 Embedded MPEG Decoder System Design Exploration

The primary motivation for the development of cycle-accurate energy consumption simulation methodology is to provide an easy way for embedded system designers to evaluate multiple hardware and software architectures with respect to performance and energy consumption constraints. In this section we will present an application of the simulation methodology to embedded MPEG decoder system design exploration. The MPEG decoder design consists of the processor, the off-chip memory, the DC-DC converter, output to the LCD display, and the interface to the source of the MPEG stream. The input and output portions of the MPEG decoder design will not be considered at this point.

Table 2: Memory Architectures for MPEG Design

Name	First Acc. (ns)	Burst Acc. (ns)	Actjve Pwr (mW)	Idle Pwr (mW)	Line Cap. (pF)	Pin Cap. (pF)	Manuf.
FLASH	80	N/A	75	0.5	4.8	10	Intel
BFLASH	80	40	600	2.5	4.8	10	TI
SRAM	90	N/A	185	0.1	8	8	Toshiba
BSRAM	90	45	365	1.7	8	8	Micron
BSDRAM	30	15	430	10	8	8	Micron
L2 cache	20	10	1985	330	3.2	5	Motorola

The ARM710a processor model running at 200MHz with 400mW active and 170mW NOP power consumption is used in all the simulations. The processor has a 32Kb, 16-way set associative, unified L1 instruction and data cache. We considered using L2 cache in addition to L1 cache. Unified L2 cache is 256Kb, 4-way set associative. Their characteristics of memory components considered are shown in Table 2. Two different instruction memories were evaluated – low-power FLASH and power-hungry burst FLASH. We looked at three different data memories – low-power SRAM, faster burst SRAM and very power-hungry burst SDRAM. Both instruction and data memories are 1MB in size. DC-DC converter specifications are shown in Figure 2. The hardware configurations simulated are shown in Table 3. The MPEG decode has been fully realized in software. Each simulation uses 12 frames running at the 30 frames/second, with two I, three P and seven B-frames.

Table 3: Hardware Configurations

Name	Instruction Memory	Data Memory	L2 cache Present
Original	FLASH	SRAM	no
L2 cache	FLASH	BSDRAM	yes
Burst SRAM	BFLASH	BSRAM	no
Burst SDRAM	BFLASH	BSDRAM	no

Figure 4 shows the amount of time each system component is active during the MPEG decode and the amount of energy consumed. The original configuration is limited by the bandwidth of data memory. L2 cache is very fast, but also consumes too much energy. Burst SDRAM design fully solves the memory bandwidth problem with least energy consumption. Instruction memory constitutes a very small portion of the total energy due to the relatively large L1 cache in comparison to the MPEG code size. The DC-DC converter consumes a significant amount of total energy and thus should be considered in system simulations. We conclude from this example that using faster and more power-hungry memory can be energy efficient.

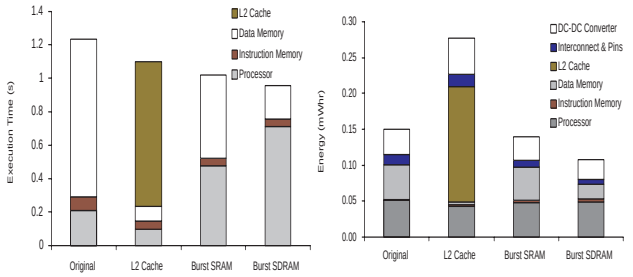

Figure 4: Execution time and energy consumption for different hardware architectures

The software architecture trade-offs are illustrated by modifying the MPEG video stream content and the decode speed. The MPEG video stream consists of three types of frames: I (intraframe), P (predicted) and B (bidirectional) [19]. P-frames are predicted from the past decoded frames. B-frames are predicted from both the past and the future frames. Four different software architectures are considered as shown in Table 4.

Table 4: Software Configurations

Configuration @ (frames/s)	I-frames (number)	P-frames (number)	B-frames (number)
IPBB @ 30	2	3	7
IP @ 30	2	10	0
IPPI @ 30	4	8	0
IPPI @ 25	4	8	0

Figure 5 shows the total energy (on the left axis) and total execution time (on the right axis) for each software configuration. Clearly the IPPI @ 30 fr/s configuration with eight I-frames and only four P-frames is the most time and energy efficient. Slower frame rate (25 fr/s) does not conserve energy.

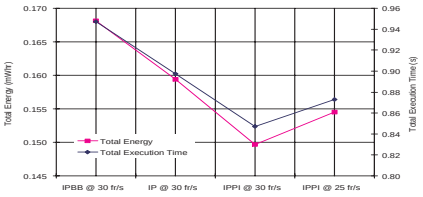

Figure 5: Energy and Exec Time for different SW configs

The analysis of peak energy consumption and the fine tuning of the architectures can be done by studying the energy consumption and the memory access patterns over a period of time. Figure 6 shows the energy consumption over time of the ARM710a processor with burst FLASH and SRAM. Peak energy consumption can reach twice the average consumption, so the thermal characteristics of the hardware design, the DC-DC converter and the battery have to be specified accordingly.

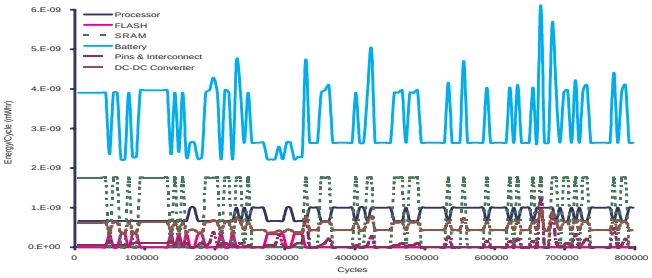

Figure 6: Energy Plot for ARM710a, FLASH and Burst SRAM

The design exploration example presented in this section illustrates how the methodology for cycle-accurate energy consumption simulation can be used to select and fine-tune the hardware and software configuration that gives the best trade-off between performance and energy consumption.

6 Conclusions

A methodology for cycle-accurate simulation of performance and energy consumption in embedded systems has been presented in this paper. Accuracy, modularity and ease of integration with the instruction-level simulators widely used in industry make this methodology very applicable to the embedded system hardware and software design exploration. We applied the methodology to the ARM Ltd. instruction-level simulator. Dhrystone benchmark has been used to verify accuracy of the energy and the performance estimates. Simulation is found to be within 5% of the hardware measurements. MPEG decoder embedded system design exploration has been presented as an example of how the methodology can be used in practice to aid in the selection of the best hardware and software configuration.

7 Acknowledgments

The authors would like to thank Mark Smith and Malena Mesarina for their help. This work was supported by the Hewlett-Packard Laboratory.

References

[1] Advanced RISC Machines Ltd (ARM), *ARM Software Development Toolkit Version 2.11*, 1996.

[2] G. Q. Maguire, M. Smith, H. W. Peter Beadle, "SmartBadges: a wearable computer and communication system," *Invited talk slides url: www.it.kth.se/maguire/Talks/CODES-980313.pdf*, 6th International Workshop on Hardware/Software Codesign, 1998.

[3] CoWare, *CoWareN2c url:www.coware.com/n2c.html* .

[4] Mentor Graphics, *www.mentor.com/codesign*.

[5] Synopsys, *www.synopsys.com/products/hwsw*.

[6] Cadence, *www.cadence.com/alta/products*.

[7] P. Landman, J. Rabaey, "Activity-Sensitive Architectural Power Analysis," *IEEE Transactions on CAD*, pp.571–587, June 1996.

[8] D. Liu, C. Svensson, "Power Consumption Estimation in CMOS VLSI Chips," *IEEE Journal of Solid-State Circuits*, vol.29, no.6, pp. 663–670, June 1994.

[9] M. Kamble, K. Ghose, "Energy-Efficiency of VLSI Caches: A Comparative Study," *10th International Conference on VLSI Design*, pp.261–267, January 1997.

[10] S. Wilton, N. Jouppi, "CACTI: An Enhanced Cache Access and Cycle Time Model," *IEEE Journal of Solid-State Circuits*, vol.31, no.5, pp.677–688, May 1996.

[11] K. Itoh, K. Sasaki, Y. Nakagome, "Trends in Low-Power RAM Circuit Technologies," *Proceedings of the IEEE*, vol.83, no.4, pp.524–543, April 1995.

[12] V. Tiwari, S. Malik, A. Wolfe, M. Lee, "Instruction Level Power Analysis," *Journal of VLSI Signal Processing Systems*, no.1, pp.223–2383, 1996.

[13] M. Wan, Y. Ichikawa, D. Lidsky, J. Rabaey, "An Energy Conscious Methodology for Early Design Exploration of Heterogeneous DSPs," *Proceedings of the Custom Intergrated Circuit Conference*, 1998.

[14] L. Benini, R. Hodgson, P. Siegel, "System-Level Power Estimation and Optimization," *Proceedings of ISLPED*, pp.173–178, 1998.

[15] Y. Li and J. Henkel, "A Framework for Estimating and Minimizing Energy Dissipation of Embedded HW/SW Systems," *Proceedings of DAC 1998*, pp.188–193, 1998.

[16] B. Kapoor, "Low Power Memory Architecutres for Video Applications," *Proceedings of the 8th Great lakes symposium on VLSI*, pp. 2–7, 1998.

[17] OZ Electronics Manufacturing, *PCB Modelling Tools url: www.oem.com.au/manu/pcbmodel.html*.

[18] A. El Gamal, Z.A. Syed, "A stochastic model for interconnections in custom integrated circuits," *IEEE Transactions on Circuits and Systems*, vol.CAS–28, no.9, pp.888–894, Sept. 1981.

[19] V. Bhaskaran, K. Konstantinides, *Image and Video Compression Standards* Kluwer Academic Publishers, 1997.