An Instruction-Level Energy Estimation Model for Embedded Systems

Chandan D R
M.Tech, 4th Semester, MITE,
Moodabidri, India.

ABSTRACT

Low power consumption has been established as the third main design target for digital systems together with performance and area. Consequently, accurate estimators of the power consumption at the system as well as the lower levels of abstraction are necessary. A large number of embedded computing applications are power or energy critical. Early work on processor analysis had focused on performance improvement without determining the power-performance tradeoffs. Recently, significant research in low power design and power estimation and analysis has been developed. In this project, the measurements taken for the development of instruction-level energy models for microprocessors are presented and analyzed. By combining energy coefficient values, the presented model became much simpler than most proposed models that can speed up the energy estimation process and therefore the development of the embedded systems. Retarget ability is another advantage of our proposed model that makes it easy to adjust the model coefficients for a new platform. The model coefficients can be calibrated by measuring the energy consumption of test programs for the new platform.

1. INTRODUCTION

An embedded system plays an important role in many areas of human life. Cell phones, PDAs, and satellites are only few examples of devices with a processor embedded in them. A large group of these systems are portable battery powered devices that have a limited source of energy. This makes the energy consumption a prominent characteristic. Hence, energy estimation during design phase of these applications helps designers in optimizing the energy consumption and the battery lifetime. Since software is responsible for a large portion of the system energy consumption, an accurate energy model is necessary for the system energy optimization [1].

Embedded software energy modelling techniques are distinguished into two main categories, physical measurement-based and simulation-based ones. In measurement-based approaches, the energy consumption of software is characterized by data obtained from real hardware. In simulation-based methods, energy consumed by software is estimated by calculating the energy consumption of various components in the target processor through simulations at different levels.

Two main reasons motivate the constant effort of the designers to reduce p o w e r consumption. First, in many cases, embedded systems are designed for mobile applications using a limited life-time battery.

Second, thermal dissipation is constrained by cost, weight and size limits. As a consequence, working temperature must be kept low enough to allow cheap packaging and/or maintain system reliability above the required level.

Current techniques for embedded system design constantly seek new improvements for maximum reusability. One common solution is platform-based design, where system platforms contain generic and specific purpose hardware devices controlled by application software executed on general purpose microprocessors. The development of the software accounts for more than 80% of the total effort of the design system [1]. Energy efficiency of software is lower than hardware. As a consequence, software execution represents an important percentage of the total energy consumption of the system. A design can be rejected because the energy consumption needed to perform software tasks is excessive. This possible drawback justifies the necessity of simulations to estimate this consumption at earlier stages of the design flow.

2. LITERATURE SURVEY

In 1994, Tiwari et al. [2] have presented an energy modelling of embedded software's executed by an embedded processor. This model includes only the energy of instructions and inters instruction energy costs.

In 1995, Chang et al. [3] have analyzed the energy consumption of a modified ARM7TDMI core prototype. However, this model does not consider the energy consumption of the Flash memory, opcode of instructions, number and value of registers, instruction and data fetch addresses or SRAM.

$$\text{Energy} = \alpha h + \beta w + \gamma$$

In 2003, Nikolaidis et al. [4] have proposed a paper, a model for in-order pipelined processors based on the model proposed by Chang et al. [3] with the inclusion of pipeline stall energy costs has been proposed peripherals.

In 2004, Kavvadias et al. [5] have improved the energy estimation model proposed by Nikolaidis et al. [4] by including the pipeline flush and pipeline stall caused by load store, multiply or branch instructions. The final model is evaluated for an ARM7 processor using real software kernels. However, the energy consumption of memories has not been considered in these works.

In 2001, Steinke [6] targets an ARM7-based platform and proposes a more detailed energy model. The total energy consumption is modelled as the sum of instruction-dependant CPU and memory energy consumption plus the activation cost of each functional unit of the processor. This model includes the same parameters as the model proposed by Kavvadias et al. [5].

In 2001, Lee et al. [7] have used a "black-box" or "stimulus-response" approach where a set of test programs are executed on the hardware platform, the energy consumption is measured, and the model parameters are extracted using the regression methods. It only estimates the energy consumption of the processor and does not cover the energy consumption of memories or peripherals.

3. METHODOLOGY

The technique presented in this work performs source code energy estimation from the operators individual energies. The operator energy is obtained by adding the energy consumed by all machine instructions executed to perform the operation.

Assuming N is the total number of C operators and control statements, the total energy E_T necessary to execute the whole application code is

$$E_T = \sum_{n=1}^{N} E_n \qquad (1)$$

where, E_n is the total amount of energy consumed related to operator 'n'.

E_n depends on how many times this operator is executed and the corresponding execution energy of each one.
In this work, operations with variables and operations with immediate values are considered different, so, in fact:

$$N = (2 * N_{op}) + N_{ctr} \qquad (2)$$

where N_{op} is the number of C operators.

N_{ctr} is the number of C control statements.

The energy consumed can be different each time the operator is executed. Depending on the data location (memory - registers) or the data size, the number of assembler instructions used can be different.
If M_n is the number of executions of operator 'n' and E_{nm} is the energy for each single operator execution:

$$E_n = \sum_{m=1}^{M_n} E_{nm} \qquad (3)$$

Therefore, the equation for E_T becomes,

$$E_T = \sum_{n=1}^{N} \sum_{m=1}^{M_n} E_{nm} \qquad (4)$$

The energy required by one operator in a single execution necessary to implement it. It can be calculated in the following way:

$$E_{nm} = \sum_{i=1}^{I_{nm}} E'_I \qquad (5)$$

Where I_{nm} is the number of machine instructions required to execute the operator 'n' in execution 'm'.

E'_i the energy of the corresponding machine instruction.

Substituting (2.5) into (2.4), the result is:

$$E_T = \sum_{n=1}^{N} \sum_{m=1}^{M_n} \sum_{i=1}^{I_{nm}} E'_I \qquad (6)$$

To simplify this mean value, by using the mean value

$$E'_{nm} = \frac{\sum_{i=1}^{I_{nm}} E'_i}{I_{nm}} \qquad (7)$$

And therefore E_{nm} becomes,

$$E_{nm} = \sum_{i=1}^{I_{nm}} E'_i = I_{nm} E'_{nm} \qquad (8)$$

As a consequence, in equation (6), the mean energy consumption per instruction can be extracted from the mean number of assembler instructions necessary to implement the operator.

$$E_T = \sum_{n=1}^{N} \sum_{m=1}^{M_n} I_{nm} E'_I \qquad (9)$$

$$E_T = E'_i \sum_{n=1}^{N} \sum_{m=1}^{M_n} I_{nm} \qquad (10)$$

This is the equation for energy calculation of operators and control statements.

3.1 Energy Estimation of Single Instruction

The total energy E_T necessary to execute the whole application code is:

$$E_T = \sum_{n=1}^{N} E_n \qquad (11)$$

Where, E_n is the total amount of energy consumed by instruction I. n is the number of instructions.

In this work, E_n is the total amount of energy consumed by instruction I, so, in fact:

$$E_n = \sum_{m=1}^{o} E_m \qquad (12)$$

Where O is the number of times instruction I appeared in the program.

E_m is the energy consumed by instruction I in the m^{th} time .

Energy consumed by instruction I in the m^{th} time is given by

$$E_m = k_m * g * E_{clk} \qquad (13)$$

The Energy of one machine cycle is given by $E_{clk} = p/f$.

Where, p is the power consumption and f is the frequency of ARM7TDMI.

Then the energy of a single instruction for respective cycle is given by,

$$E_T = \sum_{n=1}^{I} \sum_{m=1}^{o} K_m * g * p/f \qquad (14)$$

Let K_a be the average cycle of instruction cycle n.

Therefore, the energy of a single instruction is given by,

$$E_T = \sum_{n=1}^{I} K_a * g * p/f \qquad (15)$$

4. RESULTS

4.1 Energy calculation of C Controls Statements.

Control Statements	Operation	Energy (nj)
For	for (a=0 ; a<10; a++)	3.1818
	(r3=0 ; r3<10; r3++)	1.8182
If	if(a==0)	1.5909
	if(r3==0)	1.1364
While	while(a<0)	.9091
	while(r3<0)	.9091

4.2 Energy calculation of C operators.

Operators	Operation	Energy (nj)
+ operator	C=a+b	1.8182
	C=a+23	1.3637
	C=23+43	0.9091
	R3=r2+r4	0.45455
	R3=r2+23	0.45455
	R3=23+43	0.45455
= operator	C=a	1.3637
	C=23	0.9091
	C=r3	0.45455
	Mov r3, r2	0.45455

4.3 Energy estimation of each instruction.

Instruction Parameters			
Param Energy(nj)	Param Energy(nj)	Param Energy(nj)	Param Energy(nj)
ADC 1.13636	MUL 3.18185	LDM 2.11612	SMULL4.09095
LDR 1.81818	BIC 1.13636	STM 1.25001	SMLAL5.00005
STR 0.909091	TST 1.13636	SWP 3.63636	CMP 1.13636
EOR 1.13636	UMLAL5.00005	MVN 1.13636	B 1.36364
UMULL4.09095	SUB 1.13636	MLA 3.68182	AND 1.13636

4.4 Comparison of simulation based values with Measurement based values.

Instruction	Measurement based (nj)	Simulation based (nj)	Variation (nj)
ADC	1.127	1.13636	0.009
LDR	1.84	1.81818	0.022
STR	1.343	0.909091	-0.434
EOR	1.167	1.13636	-0.031
UMULL	4.254	4.09095	-0.163
MUL	2.931	3.18185	0.251
B	0.79	1.36364	0.573
TST	1.006	1.13636	0.076
UMLAL	4.818	5.00005	0.182
SUB	1.143	1.13636	0.006
LDM	1.94	2.11612	0.176
STM	1.449	1.25001	-0.190
SWP	3.593	3.63636	0.043
MVN	1.13	1.13636	0.034
MLA	3.423	4.09095	0.668
AND	1.178	1.13636	-0.416
SMULL	4.29	4.09095	-0.120
SMLAL	4.391	5.00005	0.610
CMP	0.978	1.13636	0.158
BIC	1.049	1.13636	0.087

5. CONCLUSION AND FUTURE ENHANCEMENTS

In conclusion, The measurements taken for the development of instruction-level energy models for the ARM7TDMI embedded processor are presented and analyzed. The instantaneous current drawn by the processor is measured and integrated in clock cycles to derive the consumed energy. Appropriate measuring environment and modeling methodology has been established for this purpose. The energy of an instruction is analyzed in three components. These components correspond to the pure base energy cost of the instruction, the inter-instruction cost and the effect of the instruction parameters. The values for these components were presented, analyzed and discussed. The proposed approach in modeling the software energy of microprocessors has been validated by the results.

6. REFERENCES

[1] Mostafa bazzaz, Mohammad salehi and Alireza ejlali, "An accurate instruction-level energy estimation model and tool for embedded systems," ieee transactions on instrumentation and measurement, vol. 62, no. 7, july 2013.

[2] V. Tiwari, S. Malik, and A. Wolfe, "Power analysis of embedded software: A first step towards software power minimization," IEEE Trans. VLSI Systems, vol. 2, no. 4, pp. 437–445, Dec. 1994.

[3] N. Chang, K. Kim, and H. G. Lee, "Cycle-accurate energy consumption measurement and analysis: Case study of ARM7TDMI," in Proc. ISLPED, 2000, pp. 185–190.

[4] S. Nikolaidis, N. Kavvadias, T. Laopoulos, L. Bisdounis, and S. Blionas, "Instruction level energy modeling for pipelined processors,".

[5] N. Kavvadias, P. Neofotistos, S. Nikolaidis, K. Kosmatopoulos, and T. Laopoulos, "Measurements analysis of the software-related power consumption of microprocessors," IEEE Trans. Instrum. Measurement, vol. 53, no. 4, pp. 1106–1112, Aug. 2004.

[6] K. Zotos, A. Litke, E. Chatzigeorgiou, S. Nikolaidis, and G. Stephanides, "Energy complexity of software in embedded systems," in Proc. IASTED, Jun. 2005, pp. 17–27.

[7] S. Steinke, M. Knauer, L. Wehmeyer, and P. Marwedel, "An accurate and fine grain instruction-level energy model supporting software optimizations," in Proc. Int. Workshop PATMOS, Sep. 2001.

[8] V. Konstantakos, A. Chatzigeorgiou, S. Nikolaidis, and T. Laopoulos, "Energy consumption estimation in embedded systems," IEEE Trans. Instrum. Meas., vol. 57, no. 4, pp. 797–804, Apr. 2008.

[9] S. Lee, A. Ermedahl, and S. L. Min, "An accurate instruction-level energy consumption model for embedded RISCprocessors," in Proc. 5th ACM SIGPLAN Workshop LCTES, Aug. 2001, pp. 1–10.

[10] V. Konstantakos, A. Chatzigeorgiou, S. Nikolaidis, and T. Laopoulos, "Energy consumption estimation in embedded systems," in Proc.IEEE IMTC, Apr. 2006, pp. 235 238.

[11] T. Simunié, L. Benini, G. De Micheli, "Cycle-Accurate Simulation of Energy Consumption in Embedded Systems", DAC, 1999.

[12] A. Sinha, A.P. Chandrakasan, "Joule-Track, A Web Based Tool for Software Energy Profiling", DAC, 2001.

[13] J. Laurent, E. Senn, N. Julien, E. Martin, "Power Consumption Estimation of a C algorithm: A New Perspective for Software Design", ACM LCR Conference, 2002.

[14] C. Brandolese, W. Fornaciari, L. Pomante, F. Salice, D. Sciuto, "A Multi-level Strategy for Software Power Estimation", XXX.

[15] H. Posadas, J. Adámez, E. Villar, Francisco Escuder (DS2), Francisco Blasco (DS2), "RTOS modeling in SystemC for Real- Time embedded SW simulation: A POSIX model", Design Automation for Embedded Systems, V.10, N.4, Springer, 2005.