ISSN 2250-3153

Power Aware Instruction Scheduling for Microcontrollers

K.B.R.G.T. Samarasinghe, M.M.N.N. Jayasekara, D. Elkaduwe and R. G. Ragel

Department of Computer Engineering, University of Peradeniya

Abstract- The ability to reduce power consumption of a device is attractive for several reasons. On one hand, reducing power in high end computers will reduce the cost of cooling and the performance loss due to overheating cores. On the other hand, ubiquitous battery powered devices will enjoy a longer battery life due to the reduction in power consumption. In this paper, we present a set of mechanisms that uses instruction scheduling to reduce the power consumption of RISC like microcontrollers that are common in battery powered devices. Initially, we devised a method to measure the approximate power consumption of each basic assembly instruction of the microcontroller. By a statistical analysis of the power measurement, we categorized the basic instructions into groups. Our measurements demonstrate that some instruction combinations consume more power than others which will perform the same functional operations. We leveraged this observation to rearrange the scheduling of basic machine instructions of a high level programming language, such that the final program is optimized for power. Some of our test cases demonstrate significant reduction in power consumption without any performance degradation. The resultant ideas can therefore be used by both system programmers and compiler designers.

Index Terms- energy, instruction scheduling, microcontroller, power aware scheduling, power consumption

I. INTRODUCTION

Portable or hand held devices containing embedded systems have covered a lot of areas of modern technological development. Lots of people have become addicted to popular embedded devices like mobile phones, PDAs, e-book readers, etc. Regardless of the technology or the microcontroller type (PIC, AVR, ARM, etc.) used, power management of the embedded devices is a major challenge that the designers and the manufacturers have to consider seriously.

There are two main reasons for why power management has become a major issue in designing embedded devices and why having a low power consuming microcontroller has become a plus point for embedded devices. One reason is the battery backup time of the device. As we know a lot of embedded devices are portable and are working with DC powered cells and the battery backup time enforces a limitation on the usage of the device. Having reduced or well managed power consumption can increase the battery backup time of the device and that will be a desirable quality. The second reason to have power management is the heat produced by the device.

Heat is definitely an undesired quality that affects the performance and the durability of the device, and it may add an additional cost if that excessive heat has to be reduced using a hardware solution (such as heat sinks and fans). Reducing the power consumption of embedded devices can reduce the heat generation while the device is active.

One of the main features that effects on the power consumption of an embedded device is the way in which the low level instructions are scheduled or combined together, as each machine instruction has a different power consumption value. For a given higher level language code (such as C and mikroC), there can be more than one combinations of basic hardware instructions (assembly) for a given embedded microcontroller architecture. Therefore, by selecting the optimum combination of basic assembly instructions, the total power consumption of the particular device can be reduced. When you consider an application program, converting its higher level language codes into relevant assembly instruction combination is performed by a compiler (that is, the instruction scheduler of the complier). Therefore, by optimizing a compiler to produce the optimum assembly instruction combinations targeting power of each instruction, the power consumption of the targeted embedded system can be reduced.

Given the above facts, in our project we concentrate on analyzing the comparative power consumptions of the basic assembly instruction sets of embedded microcontroller architecture and analyzing methods of using different instruction scheduling methods to minimize the power consumption.

The rest of the paper is organized as follows. In Section II, we discuss some related work under two categories: (1) Related work on obtaining power profiles of basic instructions; and (2) Related work on power aware instruction scheduling. The methodology used in our project is presented in Section III followed by results in Section IV. In Section V, we discuss the behavior of our results and concluded the paper in Section VI.

II. Related Work

There are a large amount of researches which have considered power consumption of the microcontroller systems, instruction scheduling for power optimization and power profiles of assembly instructions.

A. Related work on obtaining power profiles of basic instructions:

Many researches have been done on power profiles and power management of microcontrollers and microprocessors.

In most of these projects, microprocessors power profiles have been presented using power/energy models based on their internal architectures. Tiwari et al. have presented the power consumption of machine instructions of Intel 486DX2, a CISC microprocessor, based on an energy model [10]. Tiwari and Lee [9] have performed a set of experiments on Fujitsu- SPARClite MB86934, a 32bit RISC microprocessor and compared the results with [10]. According to them, the power model they have used in [10] and [9] can be used for any 32 bit RISC architecture.

Russel et al. [7] has followed a different approach to obtain detailed power profiles for Intel 80960 JF [3] and Intel 80960 HD [2] microprocessors. Their method includes an energy model but also involves some measurements using a digital oscilloscope. They have also presented the power variation against the clock frequency. In our work the usage of oscilloscope for power measurement was not feasible due to low sensitivity of the device.

No work among mentioned above has measured the power consumption values of each single instruction of a microcontroller in order to obtain the power profiles of instructions of a microprocessor and most of the values presented were based on estimations and power/energy models.

Even though they have provided some values based on estimations, those values cannot be used to get a clear idea on comparative power consumption of machine instructions of microcontrollers in general. The different in our approach is that, when we are obtaining power profiles of the instructions, we take actual power consumptions through power readings of a microcontroller instruction.

B. Related work on Instruction Scheduling:

When it comes to instruction scheduling of microcontrollers (or basic idea of a compiler to optimize power), there were some proposals.

The basic idea behind the work done by Steinke et al. in [8] is having a "power aware compiler" based on the energy model for ARM7TDMI-RISC microprocessor.

The authors in [6] have compared the traditional performance-oriented instruction scheduling and power-oriented instruction scheduling and based on the results they have implemented three new instruction scheduling algorithms. A cost model for the switching activities of the instruction bus for very large instructions has been presented by Lee et al. in [1]. Based on the cost model, they have implemented an instruction scheduling scheme optimized for performance. Then they have implemented two instruction scheduling algorithms to optimize the power of those instructions and compared. Kandemir et al. in [4] has presented a number of techniques for energy optimization of compilers.

We make use of the real power measurements for performing the instruction scheduling. Although we are yet to incorporate our scheduler to a compiler, we have tested our hypothesis with some smaller handwritten code snippets.

III. Methodology

We had two major steps in this project, first was measuring power consumption values of basic assembly instructions of a microcontroller system. The second step was testing possibilities and implementing models of minimizing power consumption using instruction scheduling. The methodologies we used for each one of them are discussed in detail below.

A. Measuring power consumption values of basic assembly instructions:

We used Microchip's PIC16F877a, an eight bit microcontroller with RISC architecture for our experiments. When a program or an instruction is running in the microcontroller, the only considerable amount of power dissipation takes place on the voltage-in pin (V_{DD}). Therefore by measuring voltage levels and the input current of V_{DD} pin of the microcontroller, approximate power consumption for the running instruction can be observed.

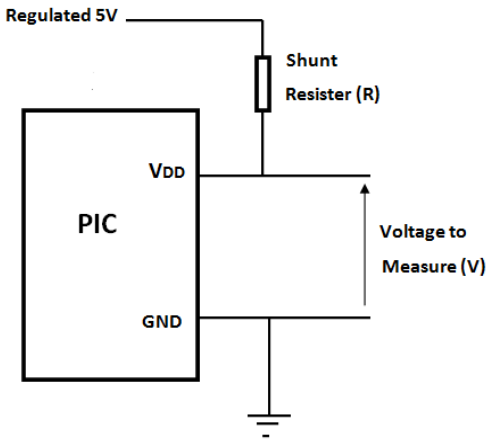

Figure 1. Model of the voltage measurement

As shown in Figure 1, when the voltage level of the voltage–in pin of the microcontroller is V_{DD} and input current isI_{DD}, the power dissipation can be given as in Equation (1).

$$P_{dis} = V_{DD} \times I_{DD} \quad (1)$$

The circuit in Figure 1 was used to measure values for calculating power consumption according to Equation 1. We measured the voltage drop across a shunt resister connected to the voltage- in (V_{DD}) pin of the microcontroller.

According to the above measurement, the power dissipation value could be represented as in Equation (2). In this form of measurement we can eliminate the need of measuring current values.

$$P_{dis} = \frac{(5-V_{DD})\times V_{DD}}{R} \quad (2)$$

The instructions are executed in very high speed (in the domain of power measurement equipment) in the microcontroller (we used a 4 MHz clock speed in our system). Therefore, the duration of a single execution is extremely small and the power value to be measured is also very small. Even if we use measuring equipment with very high sensitivity, it will not be possible to measure the power consumption values for the execution of a single instruction. Therefore we measured the voltage values (V_{DD}) by executing a large number of cycles with the same instruction and taking the average over time.

We used National Instrument's 6221-PCIdata acquisition card for measuring voltage. In our measurements, we used Single ended-Non referenced (NRSE) terminal configuration with grounded-source as in Figure2. The maximum sampling rate we could obtain was 200 kHz.

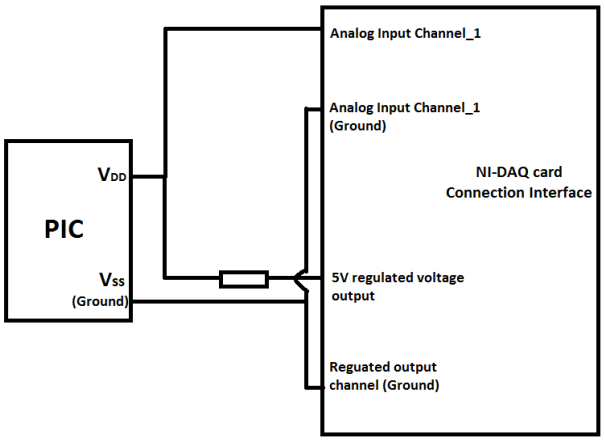

Figure 2. Circuit diagram for the voltage measurement port

We categorize 35 basic assembly instructions in PIC16F877a microcontroller. We took voltage readings for 3 sets of samples read at 200 kHz, with 100,000 samples in each sample set for each basic assembly instruction. We used DAQ tool kit 2.1 in Matlab R2008b to read and sample the voltage readings from the NI-DAQ 6221 card. After saving the sample data as Matlab matrix files, we used Statistical Analysis Toolkit in Matlab to obtain average voltage readings and the variance of the samples. Using average voltage values, we calculated average power consumption values for each basic assembly instruction.

B. Possibilities of minimizing power consumption values using instruction scheduling:

In this step we explored various methods in scheduling assembly instructions for given high -level arithmetic and logic functions, and checked whether there is any effect on power consumption by selecting different assembly instruction combination. We followed the following procedure:

Step 1: Select a higher level function (multiplication, division, sorting, etc.)

Step 2: Obtain various possibilities of implementing that higher level functions in assembly instruction combinations with exactly same operands/variables (same functionality).

Step 3: Assigning power consumption values for each assembly instruction from what we measured in Step A of the research we measured the energy consumption for each combination from step 2 using Equation (3).

$$Energy = \sum_i^{for\ all\ instructions} (Power_i\ X\ time_i) \quad (3)$$

Step 4: Assuming each instruction has same clock cycle time, we calculated the average power for each assembly combination using Equation (4).

$$Average\ Power(P_a) = \frac{Total\ Energy}{Number\ of\ instructions(i) \times execution\ time\ of\ an\ instruction} \quad (4)$$

We report the results on our experiments in the next section.

IV. Results

The measured average voltages of some important instructions are shown in Figure 3 and the calculated average power consumption using the average of the measured voltages for basic assembly instructions in PIC 16F877a are given in Table I.

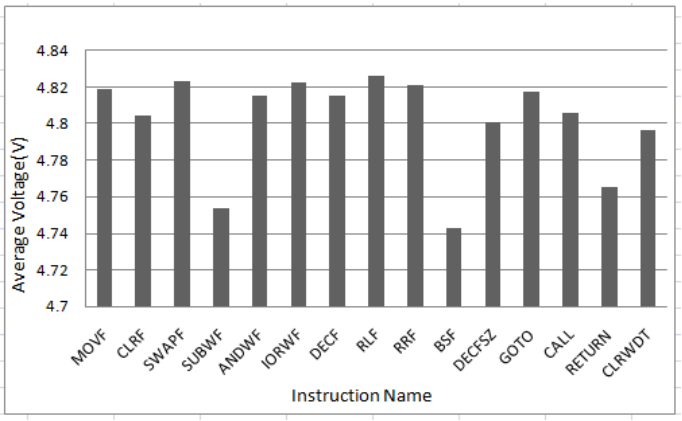

Figure 3. Average voltages of some important instructions

As it can be seen, different instructions consume different power values and this information is used to perform the power aware instruction scheduling.

Table I: Description and Calculated power values of instructions

Instruction	Description	Average power Consumption (mW)
MOVLW k	Move constant to W	8.792
MOVF f	Move W to f	8.719
MOVWFf,d	Move f to d	8.850
CLRW	Clear W	8.797
CLRF f	Clear f	9.403
SWAPFf,d	Swap nibbles in f	8.520
ADDLW k	Add W and constant	9.021
ADDWFf,d	Add W and f	8.880
SUBLW k	Subtract W from constant	8.737
SUBWFf,d	Subtract W from f	11.712
ANDLW k	Logical AND with W with constant	8.729
ANDWFf,d	Logical AND with W with f	8.875
IORLW k	Logical OR with W with constant	8.629
IORWFf,d	Logical OR with W with f	8.553
XORLW k	Logical exclusive OR with W with constant	8.773
XORWFf,d	Logical exclusive OR with W with f	8.727
INCFf,d	Increment f by 1	8.973
DECFf,d	Decrement f by 1	8.905
RLFf,d	Rotate left f through CARRY bit	8.397
RRFf,d	Rotate right f through CARRY bit	8.611
COMFf,d	Complement f	11.068
BCFf,b	Clear bit b in f	8.766
BSFf,b	Clear bit b in f	12.184
BTFSCf,b	Test bit b of f. Skip the following instruction if 0.	8.978
BTFSSf,b	Test bit b of f. Skip the following instruction if 1.	9.082
DECFSZf,d	Decrement f. Skip the following instruction if 0.	9.546
INCFSZf,d	Increment f. Skip the following instruction if 1.	8.715
GOTO k	Go to address	8.791
CALL k	Call subroutine	9.318
RETURN	Return from subroutine	11.182
RETLW k	Return with constant in W	9.214
RETFIE	Return from interrupt	8.790
NOP	No operation	8.984
CLRWDT	Clear watchdog timer	9.768
SLEEP	Go into sleep mode	7.586

f - Any memory location (register)
W - Working register (accumulator)
k - Constant
b - Bit address within an 8-bit register
d - Destination bit (If **d** = w or **d** = 0 the result is stored in the **W** register. If **d** = f or **d** = 1 the result is stored in register f.)

* The variances for the measurements were negligible and therefore not mentioned.

We considered different combinations of assembly instructions to implement the multiplication of two 4-bit numbers. Let us look at two ways of implementing the same function in assembly of PIC16F877a. In the second step of our work, we implemented multiplication of two 4-bit numbers as two different combinations (as A and B) of assembly instructions of PIC16F877a as given in Figure 4.

Assembly combination A:

```
SWAPF   N1,W;
BTFSS   N2,0
ADDWF   N2,F
RRF     N2,F
BTFSS   N2,0
ADDWF   N2,F
RRF     N2,F
BTFSS   N2,0
ADDWF   N2,F
RRF     N2,F
BTFSS   N2,0
ADDWF   N2,F
RRF     N2,F
```

Assembly combination B:

```
MOVF    X
        ADDWF   X,F
        BTFSS   Y,0
        MOVLW   0
        BTFSC   Y,1
        ADDWF   X,W
        RLF     X,F
        BTFSC   Y,2
        ADDWF   X,W
        RLF     X,F
        ETFSC   W,3
        ADDWF   X,W
```

Figure 4. PIC16F877a assembly instructions for 4bit multiplication

The energy and power analysis of both of the assembly combinations are given in Table II.

Table II: Power and energy for two assembly combination of multiplication of two 4-bit numbers

	combination A	**Combination B**
No of total instructions	13	12
Total energy (if time for each instruction = t)	114.8121 t	105.9721 t
Average Power consumption (mW)	8.832	8.831

Similarly, we analyzed different combinations of assembly instructions for some other high level functions such as, bubble sorting, square root of an integer number, analogue to digital converter, etc. Some of the results are presented in Table III and Table IV. We have not given the assembly combinations here due to the space limitation (each one of them is very long).

Table III: Power and energy for two assembly combination of analog to digital converter

	combination A	**Combination B**
No of total instructions	114	90
Total energy (if time for each instruction = t)	1039.6291 t	822.4749 t
Average Power consumption (mW)	9.120	9.139

Table IV: Power and energy for two assembly combination of analog to digital converter

	combination A	**Combination B**
No of total instructions	208	197
Total energy (if time for each instruction = t)	1937.9373t	1820.4801t
Average Power consumption (mW)	9.137	9.241

This idea of selecting the optimum combination of instructions can be used in the system programmer's level, when the code is written by the programmer. Some basic techniques can be applied by the programmer for example using iteration instead of recursion in order to minimize the number of RETURN instructions (note that the RETURN instruction has higher power consumption). Inline functions (eg: In C) can also be used. When a function is to be called, inline programming techniques copy the written code of the called function into the place in the higher level language, where the function is called, before it is compiled.

V. Discussion

When measuring the power consumption of assembly instructions, there were many limitations and short coming:

- The sampling rate we used in the experiment (200 kHz) was not enough (to have cycle accurate power measurement) as the measured instructions were running at 4MHz.
- There were interference and environmental noise during the measurement.
- Due to the reasons mentioned above, there could be errors in the voltage readings and calculated power values.
- In some of these instructions there are observable differences in power consumption values.

When looking at the different combinations of assembly instructions for higher level functions, we could find some instances with considerably lesser power/energy consumption. And in almost all the occasions where some assembly combinations take less power to operate, there wasn't any performance digression as number of clock cycles was not higher.

Having lesser average power consumption does not necessarily mean having less energy consumption. As the number of clock cycles increases, energy is higher regardless of the average power (Equation (5)).

$$Energy\ Consumption = Average\ Power \times Total\ Time \quad (5)$$

VI. Conclusion

As we have found out from our research, there are possibilities of using instruction scheduling to select the best combination among various combinations of assembly instruction for the same higher level function to reduce the power consumption. Even though it will be only minor amount of power difference per single iteration; between two different assembly instruction combinations, it can have a larger impact for long running embedded applications. The idea of selecting the best assembly instruction combination considering the power consumption can be integrated into a compiler designed for embedded microcontrollers.

References

[1] Jenq Kuen Lee, Chingren Lee and Ting Hwang, "Compiler optimization on instruction scheduling for low power". *13th International Symposium on System Synthesis* (ISSS'00), 2000.

[2] Intel Corporation. i960Hx Microprocessor User Manual, Nov. 1995. Order Number 272484-001.

[3] Intel Corporation. i960Jx Microprocessor User Manual, Sep. 1994. Order Number 272483-001.

[4] M. Kandemir, N.Vijaykrishnan, M. J. Irwin and W.Ye, "Influence of compiler optimizations on system power". *37thConference on Design Automation (DAC'00)*, pages 304–307, 2000.

[5] Microchip Technology Inc. "PIC16F87X Data Sheet 28/40-Pin 8-Bit CMOS FLASH Microcontrollers", 2001. DS30292C.

[6] A.ParikhSoontae Kim M.Kandemir, N.Vijaykrishnan and M.J. Irwin, "Instruction scheduling for low power". *Journal of VLSI Signal Processing*, 37(1):129–149, 2004.

[7] Jeffry T. Russell and Margarida F. Jacome, "Software power estimation and optimization for high performance, 32-bit embedded processors". *In Computer Design: VLSI in Computers and Processors*, 1998. ICCD '98. Proceedings. International Conference, pages 328 – 333, Oct 1998.

[8] Lars Wehmeyer Stefan Steinke, Markus Knauer and Peter Marwedel, "An accurate and fine grain instruction-level energy model supporting software optimizations". Technical report, University of Dortmund, Computer Science 12, Otto-Hahn-Strasse 16, 44221 Dortmund, Germany.

[9] Vivek Tiwari and Mike Tien-Chien Lee, "Power analysis of a 32-bit embedded microcontroller". *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, pages 141 – 148, Sep 1995.

[10] Sharad Malik, Vivek Tiwari and Andrew Wolfe, "Power analysis of embedded software: a first step towards software power minimization". *Very Large Scale Integration (VLSI) Systems, IEEE Transactions*, 2:437 – 445, December 1994.

Authors

First Author – K.B.R.G.T. Samarasinghe, BSC Eng. (Hons), Department of Computer Engineering, University of Peradeniya., Email:gihants@gmail.com

Second Author – M.M.N.N.Jayasekara, BSC Eng. (Hons), Department of Computer Engineering, University of Peradeniya., Email:nalakae06105@gmail.com

Third Author - D. Elkaduwe, BSc Eng (Peradeniya), PhD (UNSW), Department of Computer Engineering, University of Peradeniya, Email: dhammika@ce.pdn.ac.lk

Forth Author – R. G. Ragel, BSc Eng (Peradeniya), PhD (UNSW), Department of Computer Engineering, University of Peradeniya., Email: roshanr@ce.pdn.ac.lk