International Journal of Soft Computing and Engineering (IJSCE)
ISSN: 2231-2307(Online), Volume-1, Issue-1, March 2011

Efficient Coding Mechanism for Low Power Consumption in Wireless Programmable Devices

Kavita Deshmukh, Deepshikha Patel, Nitesh Gupta, Shiv Kumar

***Abstract*—Due to the continuous advancement in technology embedded devices are playing important role in everyone's day to day life. Everyone is moving towards wireless embedded systems, but there is important concern about power consumption in such devices. While designing a lot of care has to be taken especially in power optimization because there is no regular power supply in this kind of devices. Power optimization can be done either by changes in hardware components or changes in software programs developed for various applications. Changes in hardware, is vendor dependent and only software level changes can be done after the manufacturing of the device. This paper represents software level methods for optimizing the power in ireless embedded devices. Previously loop optimization methods where used and tested to reduce the power consumption by some degree, similarly loop unrolling and loop alignment were also used in the application to improve the performance in terms of power. In this paper we are combining some software level methods like nested switches, no of parameters , no of local variables , size of constructor, data type etc. with the loop unrolling. Our coding methodology will improve the program for consumption of the power while executing various instructions under the embedded systems.**

***Index Terms*— embedded systems, power optimization, software level methods, loop optimization, coding mechanism.**

I. INTRODUCTION

Advancement in technology and consumer's demands has transformed personal communication devices, from simple voice call terminals, to rich multimedia applications platforms. New generation's mobile phones provide various services including: internet access, video teleconferencing, global positioning, and high quality audio and video, as well as outstanding gaming capabilities. All these new features push the computing power requirements of mobile phones to the level of desktop computers. In contrast to wired devices However, the lack of a continuous power supply, poses tight limits in the overall power consumption. The limited battery lifetime has always been bottleneck when it comes to the development of improved portable electronic products [1]. In addition, constraints in the size and weight of mobile phones prohibit the use of heavy and large battery packs as power sources. There are severe restrictions on size and power consumption of wireless embedded devices. Battery technologies are improving continuously but there is parallel improvement in embedded devices at the hardware and software level. Embedded market is influenced with low power consumption, long battery life and light weight battery. As the use of wireless embedded device is increasing exponentially, there is a need of intelligent power optimization technique to serve better services to the end user. Hardware level changes dependent on device vendors and is not very universally accepted but software level design may be significantly used to optimize the power of the device without any change in hardware of the device. There are many choices of software implementation at the operating system level and application level which may affect the power efficiency of the wireless embedded devices. It has been seen from the studies that operating system calls, power efficient compile code and memory access pattern play important role in determining the overall power dissipation of the system [2].

In this paper our major concern is to study the methodologies for the power optimization which is applicable to set of code segments in a high level language. There are many methods to optimize the power of the device at the source code level changes. Loop unrolling is a great feature when writing fast code. The goal of loop unrolling is to increase a program's speed by reducing (or eliminating) instructions that control the loop, such as pointer arithmetic and "end of loop" tests on each iteration. Loops can be re-written instead of a repeated sequence of similar independent statements eliminating this overhead. Some optimization techniques developed for embedded software to optimizing the code like Adjust structure size to power of two, place frequent case labels first, break big switch statements into nested switches, minimize local variables, declare local variables in the inner most scope, reduce the number of parameters, consider locality of reference for code and data, prefer int over char and short, define lightweight

Manuscript received March 25, 2011.

Kavita Deshmukh, Information Technology, Technocrats Institute of Technology, Bhopal, M.P., India, (kavitadeshmukh27@gamil.com).

Prof.Deepshikha Patel, Information Technology, Technocrats Institute of Technology, Bhopal, M.P., India, (23.deepshikha@gamil.com).

Prof.Nitesh Gupta, Head of dept. Information Technology, Technocrats Institute of Technology, Bhopal, India, (gupta_nitesh81@yahoo.com).

Prof.ShivKumar, M.Tech. Coordinator, Dept. of Information Technology, Technocrats Institute of Technology, Bhopal, M.P., India, (shivksahu@rediffmail.com).

constructors etc. This paper also deals with the combined approach of power optimization of the wireless embedded device. Unrolling when mixed with the above techniques result into the new method of optimizing the source code. The above approach has been implemented and tested with various kind of application like text based application, multimedia application.

II. LOW POWER DESIGN CONCEPTS

Designing the low power system impact all stages of design process. Software is the major concepts of today's wireless system and its role is continuously growing [3]. Power evaluation of the device gives the designer, the ability to optimize the source code in terms of power. Compiler also performs many optimizations but it is not always better than the human. Programmers understand the nature of the program very well and he has additional knowledge of source code that the compiler lacks. Optimization like moving constant expression outside the loop, strong variable in registers using inline function and unrolling etc. can be used to optimize the power. Low power design techniques for energy-efficient embedded systems addresses the development and validation of co-synthesis techniques that allow an effective design of embedded systems with low energy dissipation [4][5]. They provides an overview of a software-level design flow, illustrating through examples how system performance is influenced at various steps of software development including allocation, mapping, and scheduling. This special emphasis upon software-level techniques for embedded device that contain voltage scalable processors [6], which can dynamically trade off between computational performance and power consumption.

III. RELATED WORK

A loop optimization represents a sequence of functions that affect the various aspects of the cache code model. For example, in loop reversal, the direction in which a loop traverses its iteration range is reversed. Our optimization method have an impact function that describes the loop header is changed to the new traversal order. We have developed optimization method for loop interchange, loop tiling, loop reversal, loop fusion, loop distribution and loop unrolling.

There has been considerable effort to develop source-to-source transformation methods that restructure loop constructs to expose possibilities for parallelism. Most published loop restructuring methods are designed for countable loops, where the iteration count can be determined without executing the loop. One such method, called loop unrolling, is designed to unroll FOR loops for parallelizing and optimizing compilers. To illustrate, consider the Following loop:

```
for (i = 1; i <= 60;i++)
a[i] = a[i] * b + c;
```

This FOR loop can be transformed into the following equivalent loop consisting of multiple copies of the original loop body:

```
for (i = 1; i <= 60; i+=3)
{
a[i] = a[i] * b + c;
a[i+1] = a[i+1] * b + c;
a[i+2] = a[i+2] * b + c;
}
```

The loop is said to have been unrolled twice, and the unrolled loop should run faster because of reduction in loop overhead. Loop unrolling was initially developed for reducing loop overhead and for exposing instruction level parallelism for machines with multiple functional units. By increasing the size of the loop body, the instruction scheduler can often produce a shorter schedule for the unrolled loop. Unlike FOR loops operating on arrays, which can be unrolled simply by modifying the counter and termination condition of the loop as illustrated above. Knowing a loop's trip count during the various phases of a program's execution can enable more effective application of loop unrolling. During the program phase where the loop trip count is high, the loop should be unrolled as the benefit will be high. The cost in terms of code space is justified. During the program phase where the loop trip count is low or zero, the loop should not be unrolled as the benefit is low and potentially negative if unrolling causes some frequently executed code to be evicted from the instruction cache. Furthermore, having accurate information at run time not only determines what optimizations to apply and when, but it also enables more effective application of certain beneficial optimizations. For example, knowing which branch of an if then- else statement is executed more frequently at different points in a program execution can enable more effective code placement.

To explore the spectrum of static to dynamic optimizations, we have begun to design, implement, and evaluate an innovative optimization paradigm and algorithms that we predict can achieve significant improvements in run time performance. Embedded software often runs on processors with limited computation power, thus optimizing the code becomes a necessity [7]. In this case we will explore the following optimization techniques developed for real-time and embedded systems.

1. Adjust structure sizes to power of two
2. Place frequent case labels first
3. Break big switch statements into nested switches
4. Minimize local variables
5. Declare local variables in the inner most scope
6. Reduce the number of parameters
7. Consider locality of reference for code and data
8. Prefer into over char and short
9. Define lightweight constructors etc.

IV. PROPOSED WORK

Power analysis techniques have been proposed for embedded software based on source code level changes and simulation of the underlying hardware. Techniques to improve the efficiency of software power analysis through statistical profiling have been proposed [8] We have

proposed following method for optimization of power consumption in wireless embedded device [9][10].

Loop unrolling is a technique which is used to optimize the program execution speed in terms of binary size. It could be done either by programmer or by optimizing compiler. In this methodology program's speed is increased by reducing instruction that control the loop. Loops can be rewritten with the help of new arrangement of sequence of statements. The advantage can be taken if two statements which are independent of each other can potentially be executed in parallel. The optimizing complier will sometimes perform unrolling automatically or upon request. Only unrolling may cause various problems like decrement of the code readability, missing of instruction cache, increasing the size of program code and increasing the number of temporarily variables. There are various other power optimization techniques like Adjust structure size to power of two, place frequent case labels first, break big switch statements into nested switches, minimize local variables, declare local variables in the inner most scope, reduce the number of parameters, consider locality of reference for code and data, prefer int over char and short, define lightweight constructors etc. can be used with unrolling to optimize the power consumption of embedded device. In this paper, we are proposing the combination of above optimization technique with unrolling to rearrange the source code written by the programmer. Our proposed work with these methods include

A. Unrolling using local variable in the inner most loop

To reduce the overhead of calls do not declare all the local variables in the outermost function scope. There would be better performance if local variables are declared in the inner most scope.

Following segment of code in java depict the same.

```
RecordStore rs;
for(int i=1;i<=rs.getNumRecords();i++)
{
for(int j=1;j<=i;j+=2)//unrolling
{
byte[] record1=rs.getRecord(i);//local variable
byte[] record2=rs.getRecord(i+1);//local variable
-------------
-------------
-------------
}
}
```

B. Unrolling with light weight constructor

As far as possible, keep the constructor light weight. The constructor will be invoked for every object creation. Keep in mind that many times the compiler might be creating temporary object over and above the explicit object creations in your program. Thus optimizing the constructor might give you a big boost in performance if it is created while unrolling the loop. If you have an array of objects, the default constructor for the object should be optimized first as the constructor gets invoked for every object in the array and this should be done while unrolling the loop.

C. Unrolling with nested switches

To reduce the number of comparisons being performed, judiciously break big switch statements into nested switches. Put frequently occurring case labels into one switch and keep the rest of case labels into one switch and keep the rest of case labels into another switch which is the default leg of the first switch.

Following segment of code in java depict the same.

```
//unrolling each record; //execution is fast, but
//retrieve three records at time; //there must odd set of records in the store.

RecordStore rs;
for(int i=1;i<=rs.getNumRecords();i+=3)
{
byte[] record1=rs.getRecord(i);
byte[] record2=rs.getRecord(i+1);
byte[] record2=rs.getRecord(i+2);
 String str1=new String(record1);
String str2=new String(record2);
 String str3=new String(record3);
switch(str1)
{
case 'str2': switch(str3)
{
case '1': ------- break;
case '2': ------- break;
case '3': -------
------- break;
--------
--------
--------
}
break;
case 'str3':
switch(str2)
{
case '1': -------
break;
case '2': -------
 break;
case '3': ------
------
------
break;
-----
-----
}
break;
--------
--------
}
```

V. POWER MEASUREMENT TOOLS

Mobile applications are typically developed using development kits and emulator and later tested on a real device. In this paper we are mainly concerned about correct behavior of the application[11][12]. Our application is energy conscious [13] hence powerful tool is required to measure the power consumption by device with respect to time when the application is tested. In this

paper, we are using power consumption profiling application to for creating energy efficient application with the help of techniques we have already mentioned in the above section [14]. Our main objective is to show the graphical result of the total power consumption of the device on which our energy efficient application is running. This profiling tools work well on latest Nokia S60 phones and no additional hardware is needed [15][16].

According to our analysis, the built in measurements are quite accurate and the remain very accurate with respect to the external references. Our measurements are concern with current and voltage. In our application the graphical result would be shown in the screen when the application would be executed by the user. We have created a power consumption profiling application to assist developers in creating energy-efficient applications and squeeze more use time from the battery. Power consumption measurements are implemented by observing battery current and voltage values. Battery current values are truly average values because the hardware integrates current consumption in the analog domain over the entire measurement period. An important characteristic for any kind of instrumentation is that it should introduce minimal disturbance to the actual measurements. This characteristic must apply particularly for an energy profiling application [17]. Graphical power measurement analysis on a small phone display must be very easy in use. This requires careful user interface (UI) design on a device with restricted display and input capabilities [18].

VI. RESULT ANALYSIS

We have examined the effect of use of unrolling method along with various software techniques with multimedia application like execution of MIDI file, wave audio file, video MPEG, animated GIF. In first phase of analysis various multimedia applications have been tested without using unrolling and other techniques. Then in a second phase analysis various multimedia applications have been tested using unrolling and other techniques [19][20].

In this paper, we have used source level code optimization techniques for various multimedia applications using unrolling with the combination of various optimization software techniques and compared the result of power consumption with the standard multimedia application without any changes. It is concluded that overall power savings achieved with source level code optimizations gave better results than those from simple multimedia application. In contrast to the previous work in literature, this work

addresses use of unrolling methods with combination of various software level techniques to evaluate the power consumption of a wireless embedded system [21]. In order to make this analysis, we designed an experiment for understanding the effect of usage of unrolling along with various software level methods for power optimizations. The graphical result is generated and shown by the energy profiling tool [22].

Snapshot of the result

Table 1 Values in MIDI executed file in mA

Time (in Sec.)	Normal Data	Optimized Data
0	135	139
15	127	123
30	119	124
45	119	112
60	112	112
75	116	114

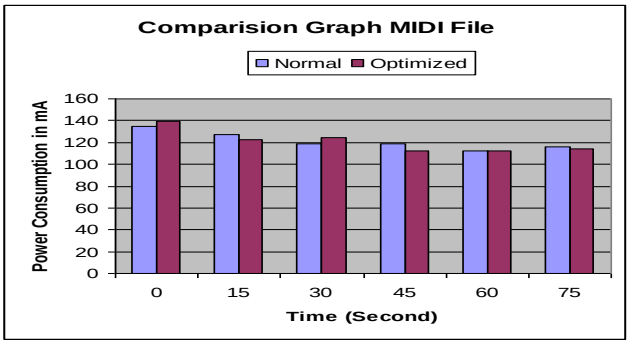

Figure 1: Power Comparison Graph in mA

Figure 1 shows the comparison of power consumption in terms of mA with respect to time for normal multimedia application and optimized multimedia application (proposed by us) in this figure it is clear that on the average, the power consumption is low in our optimized multimedia application as compared to normal multimedia application. Data have been captured from energy profiler for different applications. In a similar fashion other graphs and tables shows the result of the power consumption in terms of some other parameter s of power consumption. Since the objective of this work is to evaluate the effect that code optimizations in power consumption in embedded systems, we followed an experimental approach for this study [23]. A set of loop-oriented optimizations were selected and applied to certain segments of code. Power consumption on the embedded devices was measured while running the application. Finally, obtained results were analyzed using statistical and showing graphs.

VII. CONCLUSION

Wireless Embedded devices are playing important role not only in communication but also in day to day working of life. Although wireless embedded devices are full of constraints in terms of power, display, computation and storage. Hardware as well as software design impact the power consumption of the device. Hardware level design is specific to vendor but software design issues can be solved by developers. Loop transformation method like loop unrolling with the combination of various techniques like nested switch, lightweight constructor were experimented in a multimedia application to measure the consumption of the power of the device. Results were taken with the help of energy profiling tools. It is clear from the result at there is improvement in terms of power saving when loop

transformation method was used with code optimization techniques.

REFERENCES

[1] T. Simunic, G. De Micheli, and L. Benini, "Energy efficient design of battery-powered embedded systems," Proc. Int. Symp. Low Power Electronics and Design, Aug. 1999.

[2] J. L. Da Silva, F. Catthoor, D. Verkest, and H. DeMan, "Powe exploration for dynamic data types through virtualmemory management refinement," Proc. Int. Symp. Low Power Electronics and Design, Aug. 1999.

[3] L. Benini and G. De Micheli, "System level power optimization: Techniques and tools," Proc. Int. Symp. Low Power Electronics and Design, Aug. 1999.

[4] A. Smailagic, D. Siewiorek, M. Ettus, "System Design of Low Energy Wearable Computers with Wireless Networking," Proc. IEEE CS Workshop on VLSI 2001, Orlando, FL, April 2001, pp. 25-30.

[5] Y. Li and J. Henkel, "A framework for estimating and minimizing energy dissipation of embedded HW/SW systems," Proc. Design Automation Conf., pages 188-193, June 1998.

[6] C-T. Hsieh and M. Pedram, "Micro-processor power estimation using profile - driven program synthesis," IEEE Trans. on Computer Aided Design, Vol. 17, No. 11, pages 1080-1089, Nov. 1998.

[7] H. M. Wang , H . S. Choi, and J. T. Kim, "Workload- based dynamic voltage scaling with the qos for reaming video," in Proc. 4th IEEE International Symposiumon Electronic Design, Test & Applications, 2008, pp. 236–239.

[8] K. Holger, " An Overview of Energy - Efficiency Techniques for Mobile Communication Systems," TKN Technical Report, 2003.

[9] Cristina Silvano, "Power Estimation and Optimization Methodologies for Digital Circuits and Systems". Ph.D. Thesis, University of Brescia, 1998.

[10] T. Pering, V. Raghunathan, and R. Want. Exploiting Radio hierarchies forpower – efficient wireless device discovery and connection setup. In Proceedings of The IEEE International Conference on VLSI Design, January 2005.

[11] P. Rodriguez, R. Chakravorty, J. Chesterfield, I. Pratt, and S. Banerjee. Mar: A commuter router infrastructure for the mobile internet. In Proceedings of the Second International Conference on Mobile Systems, Applications, and Services, Boston, MA, June 2004.

[12] G.Chinn, S.Desai, Eric DiStefano, K. Ravichandran, and S. Thakkar. Mobile PC platforms enabled with Intel Centrino mobile technology. Intel Technology Journal, 7(2), May 2003.

[13] T. Simunic, L.Benini,andG. De Micheli, "Cycleaccurate simulation of energy consumption in embedded systems," Proc. Design Automation Conf., pages 867-872, June-1999.

[14] K. Xu and C. S. Choy, "A power - efficient and self adaptive prediction engine for h .264/avc encoding," IEEE Transactions on Very Large Scale Integration Systems, vol. 16, no. 3, pp. 302–313, 2008

[15] A. Whitaker, R. S. Cox , M. Shaw, and S. D. Gribble. Constructing services with interposable virtual hardware. In Proceedings of the First Symposium on Networked Systems Design and Implementation (NSDI'04), San Francisco, CA, March 2004.

[16] C. E. Jones, K. M. Sivalingam, P. Agrawal, and J. C. Chen. "A survey of energy efficient network protocols for wireless networks", Wireless Networks, 7(4):343–358, July 2001.

[17] J. Flinn and M. Satyanarayanan. Managing battery lifetimewith energy - aware adaptation. ACM Transactions on Computer Systems (TOCS), 22(2), May 2004.

[18] J . Ahn, J. hi Min, H. Cha, and R. Ha,"A power management mechanism for handheld systems having a multimedia accelerator," in Proc. Sixth Annual IEEE International Conference on Pervasive Computing and Communications, 2008, pp. 663–668.

[19] G.Manjunath, V. Krishnan, T. Simunic, J.Tourrilhes, A. McReynolds,D.Das,V. Srinivasmurthy, A. Srinivasan: "Smart Edge Server – going beyond access point," WMASH'04

[20] Z. Cao, B. Foo, L. He, and M. van der Schaar, "Optimality and improvement of dynamic voltage scaling algorithms for multimedia applications," in Proc. 45th annual Design Automation Conference, Anaheim, California, June 2008, pp. 179–184.

[21] M. Anand, E. B. Nightingale, and J. Flinn. Self- tuning Wir elessnetwork power management. In Proceedings of the 9th ACM International Conference on Mobile Computing and Networking MobiCom'03), San Diego, CA, September 2003

[22] T Simunic, W. Quadeer, G. De Micheli: "Managing heterogeneous wireless environments via Hotspot servers, " MMCN'05

[23] E. Akyol and M. van der Schaar, "Compression- aware energy optimization for video decoding ," IEEE Transactions on Circuits and Systems for Video Technology, vol. 18, no. 9, pp. 1300– 1306,2008