International Journal of Engineering & Technology, 7 (4.10) (2018) 1037-1040

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Power Optimization in System Level Using Profiling Technique

Karthik.S[1]*, Balaji T.S[2], Nihitha S.S[3], K.Priyadarsini[4]

[1]*Assistant Professor (Sr.G),Dept of ECE,SRMIST Vadapalani*
[2]*Assistant Professor, Dept of ECE,SRMIST Vadapalani*
[3]*UG Scholar, Dept of ECE,SRMIST Vadapalani*
[4]*Research Scholar,VISTAS,Pallavaram*
**Corresponding author E-mail: skarthikvit@gmail.com*

Abstract

Power optimization has become an essential and major source of concern at the system level design. With the shrink in transistor length and the attractiveness of handy electronic devices, power dissipation has become a serious issue. System level power optimization technique gains more popularity since many techniques can be applied in reduction of power. Although, there are numerous techniques to reduce power dissipation, simpler methods has not been implemented on a system level. In this paper we use profiling technique, a process which helps one to know which portion of the function takes more time compared to other during the simulation. The portion which takes large time for simulation is port mapped to the FPGA while the rest is assigned to the processor. By this technique we can see considerable amount of power savings.

Keywords: *Profiling; Vivado; Xilinx SDK; Zynq; FPGA*

1. Introduction

In most of the embedded systems the power consumption is a major task. While designing an efficient embedded system, it is required to minimize the power consumption by reducing the dissipation of energy in an all aspects and stages of the design process. During the power efficient designing of an embedded system, it is taken care that the system performance and the service quality could not affect. Various levels of design process carried out under different methods of power optimization techniques. The hierarchy levels of optimization techniques from circuit level to application level. With the help of new innovation techniques the power management expanded at every levels of optimization of power. VLSI has been derived from increasing usage of many electronic devices. Increasing the performance of the system and high clock rates are the important parameters while designing the optimized device. Along with the performance and cost, the power consumption is taken as a third parameter.

In an efficient power management system, it should not keep on depends on the source rather it tries to improve the efficiency of the system. Along the power consumption, the peak power consumption also an important concern. The power reduction technique can be carried out without affecting the circuit or device level structures.

In CMOS circuits the major power consumption at the input and output levels when compared with the state changes. Out of which the input levels is a major concern because of the signal transition occurs at the input levels due to overlapped conductance of PMOS and NMOS.

The word optimization is related with design of low power systems while active power minimization involves each and every components magnitude levels of power.

The power reduction can be obtained at the logic and circuit level as transistor sizing, transistor reordering, logic gate restructuring as well as architectural and system level also. For making dynamic power consumption the transistor size can be reduced but it has to be monitored its tolerable delay level. Sometimes, rearranging of the transistor will minimize the switching activity which in turn reduces the power consumption. These are some examples for power reduction techniques in circuit level.

Assigning various tasks at the same time interval, a group of instructions and parallel processing are the optimization of power at the architectural level. At the application level initially it tries to optimize by pipelining but the data crowd will force them to parallelism.

Out of the several power optimization techniques system level will make greatest impact on reduction of power. While considering the system level it is need to concentrate which dissipates the energy much software or hardware. In processor based system, the most power dissipation impact is depends on software [1]. While concentrating on high level languages we are ignoring the power efficiency in the sense of hardware. CPU is the major part of a computer system. Because all kind of tasks is performed through it. Its architecture and operating mechanism are considered while system or software level.

Because of software is stored in the external memory which needs additional power requirement to access unlike the cache memory and storage registers which is available internally. By means of load capacitance the memory system is utilized which means that it is associated with charging and discharging current [2]. An almost major portion of the power that can be consumed by the memory and it is more intensive in DSP applications like video processing [3]. Branching in a system level is an important factor for dissipation of power. Normally execution of a particular instruction will be carried out in a 5 stage basic model or lesser with the introduction of newer technologies. This pipelining process is an efficient process for an instruction execution sequentially. When the code is utilizes the branch instructions then it needs push a certain data before it transfers the control and retrieve it

back to take over the control back. The branch instruction may have right or wrong prediction. A penalty will be applied when a wrong predictor has been employed and this penalty is applied only in the case of mis-prediction [4]. The same CPU will dissipates the power depending upon the type of instructions [5].

2. Previous works

In Memory access reduction technique, the complex data can be handled in C/C++ with the help of pointers. Usage of pointer is efficient way but consumes more power. Whenever pointer is initialized then it is associated with memory access. When the pointer is used to access a variable x via variable a then it need to access memory through pointer chain. Instead a temporary pointer "pt" is used to access the required variable x frequently. [6] Qiaing Tong, Ken Choi et al, given that temporary pointer in the place of pointer chain produces the power saving up to 28.3965%.

Function is also a power consuming. Because each and every time, the function is called then the data access the memory space by means of pushed the data in to stack and retrieve it when its task has been completed by means of a pop instruction. The argument passing inside the function which is also the power consuming. The power optimization can be achieved through well designed function. Sometimes the reduced power consumption can attained through MACRO. It means that instead of calling as a function, the target code will be deployed during the compiling time. It will not include function cost because the code is not executed as a function. The average power reduces by 3.6887 percent in this process. The two types of function argument passing are pass- by-value and pass-by-reference. The second one is much more efficient in argument passing as well as time and power consuming. [6] Qiaing Tong, Ken Choi et al observed power saving by 21.2793 % .

In branch prediction module technique, the modern CPU will suffer from wrong branch prediction. Because of penalty of mis-prediction will take longer time to recover from mis-prediction state. Loop unrolling and combining techniques were used in the place function call. The power savings in this process are 14.0975% and 35.1535% respectively.Out of the many ALU, one at least take part for the arithmetic operation in all CPUs. For better reduction of power consumption it is a good habit that the declaration of an operand as a data which will fits in ALU size. The result shows that the function with 'int' type operand has 15.944% power reduction. In an iteration loop styles, the loop is comparing with 0 instead of non 0 produces a power saving of 12.6985%.

3. Insight to Vivado tool and Zybo board

Vivado Design Suite software [7] aimed to increase productivity with respect to designing, integrating and implementing. It is produced by Xilinx for mainly for synthesis and performs analysis of RTL design. It replaces Xilinx ISE with additional features for SOC development and high-level synthesis. With the Vivado Design Suite, you can accelerate the design, methodically optimize for multiple and parallel design metrics. Vivado permits the developers compile their designs, execute timing analysis, observe RTL diagrams, simulate a design according to a number of constraints, and configure the target device with the programmer. The main aim of this tool is to integrate more system functions with less number of parts, thereby increasing the overall system performance, and lessen system power consumption. This tool has been successfully used in the verification. After the explanation of this method one can find the effective and potential usage of this method in testing.

There are various components in Vivado design suite: a high Level compiler permits C,C++ and System C programs which can be directly targeted into Xilinx devices without converting or creating RTL.[11] A compiled-language simulator that supports diverse language, TCL scripts, encrypted IP and improved verification. A IP Integrator let's engineer to rapidly integrate and configure IP from the huge Xilinx IP library [10].

4. Zynq Architecture

The Z-7010 shown in Fig.1 is based on the Xilinx All Programmable System-on-Chip (AP SoC) architecture shown in fig1 [5], which tightly integrate a dual-core ARM Cortex-A9 processor with Xilinx 7-series Field Programmable Gate Array (FPGA) logic. When coupled with the rich set of multimedia and connectivity peripherals available on the ZYBO, the Zynq Z-7010 can host a whole system design.[9] The on board memories, video and audio Input/output, USB with dual role, Ethernet and SD slot will have your design fueled up with no extra hardware needed.[8] Additionally, six Pmod connectors are accessible to put any design on an easy development path.

5. Profiling Based Power Optimization

Profiling is achieved by instrumenting either the program source code or its binary executable form using a tool called a profiler (or code profiler). Profilers may use a number of different techniques, such as event-based, statistical, instrumented, and simulation methods. Program analysis tools are extremely important for understanding program behavior. Computer architects need such tools to evaluate how well programs will perform on new architectures. Software writers require tools to analyze their programs and identify critical sections of code. Compiler writers often use such tools to find out how well the instruction scheduling or branch prediction algorithm is performing and progressing. The steps involved in profiling are shown in Fig.2.

6. Vivado Power Analysis

The Vivado power analysis tool executes power estimation through all phase of the flow. Power analysis can be done after synthesis, placement, routing. It is mainly accurate after post-route since the exact routing resource and logic are available. The Summary of power report is obtained after every stage.Vivado helps you with environmental settings and activity factor and also helps in indentifying power consuming resources in the design. Fig.3 shows the board connections in Zybo and Fig.4 shows the Vivado power analysis flow.

7. Power Optimization Results

Suitable application project with necessary IP is created and the environment is built which is shown in fig4 and then targeted on the board and after choosing the appropriate C/C++ compiler the application is run with appropriate settings. Then we obtain the profiling results stored in gmon.out in debug column which is shown in Fig.5 and Fig.6. With the results obtained one can decided the function that has to be hardware accelerated

Table 1: Power analysis of various designs

Circuit Design name	Function identified using profiling and mapped onto FPGA as IP	Power before profiling in mw	Power after profiling in mw
Hamming Code	Min_sendbyte()	8.23	5.36
MAC unit	Mull_vt()	22.33	15.223
FIR Filter	Fir_soft()	10.2	5.97
AES	Aesrec()	9.2	3.27
ALU	Uarmes()	32.3	22.33

8. Discussion and Conclusion

After the profiling is being done, we can see the following results are observed under gmon.out file. This gives an insight about the functions that extract a longer time. To have a deeper view of the time taken by the functions, Debug option can be chosen and the application can be run in the Debug format. In the Debug format we can see clearly the inclusive and exclusive timing i.e. the time taken by the main function with its child functions and the time taken by the child functions alone respectively. The following table1 shows the power consumption of various circuit designs before and after profiling.

References

[1] Kaushik Roy , Mark C. Johnson, Software design for low power,Low power design in deep submicron electronics, Kluwer Academic Publishers, Norwell, MA, 1997.

[2] Harris, E.P., Depp S.W., Pence W,E., Kirkpatrick S. et ai,"Technology directions for portable computers" Proceedings of the IEEE, 83(4), 636-657.

[3] DeFreef E., Gatthoor F. Deman H. "Memory access coalescing:Atechnique for elimilating redundant memory access" ACM SIGPLAN Notices, 26(6).

[4] John L. Hennessy, David A. Patterson. "Computer Architecture: A Quantitative Approach"3rd edition. Morgan Kaufmann. 2003.

[5] Mike Tien-Chien Lee, Vivek Tiwari, Sharad Malik, MasahiroFujita. "Power Analysis and MinimizationTechniques forEmbedded DSP Software", IEEE Trans. on Very Large Scale Integration Systems, Vol. 5, No. 1 , March 1997.

[6] Qiaing Tong, Ken Choi, Jun Dong Cho, "A Review on System Level Low Power Techniques", IEEE Trans. On SOC Design conference, Nov 2014.

[7] Karthik.S, Dr.S.Saravanakumar, "Parallel HDL Simulation Using Heterogeneous FPGA Architectures", International Journal of Applied Engineering Research, ISSN 0973-4562, Vol.10, No.20, 2015.

[8] Lam, William K. "Hardware Design Verification: Simulation and Formal Method-Based Approache",. Prentice Hall, 2005

[9] Karthik.S, Dr.S.Saravanakumar, "Comparative Study of Homogeneous and Heterogeneous Processor in FPGA For Functional Verification" International Journal of Applied Engineering Research, ISSN 0973-4562, Volume 10, No.17, 2015.

[10] https://www.xilinx.com/support/documentaion/sw_manuals/edk10_est_rm.pdf

[11] https://www.xilinx.com/publications/prod_mktg/vivado/Vivado_9_Reasons_Backgrounder

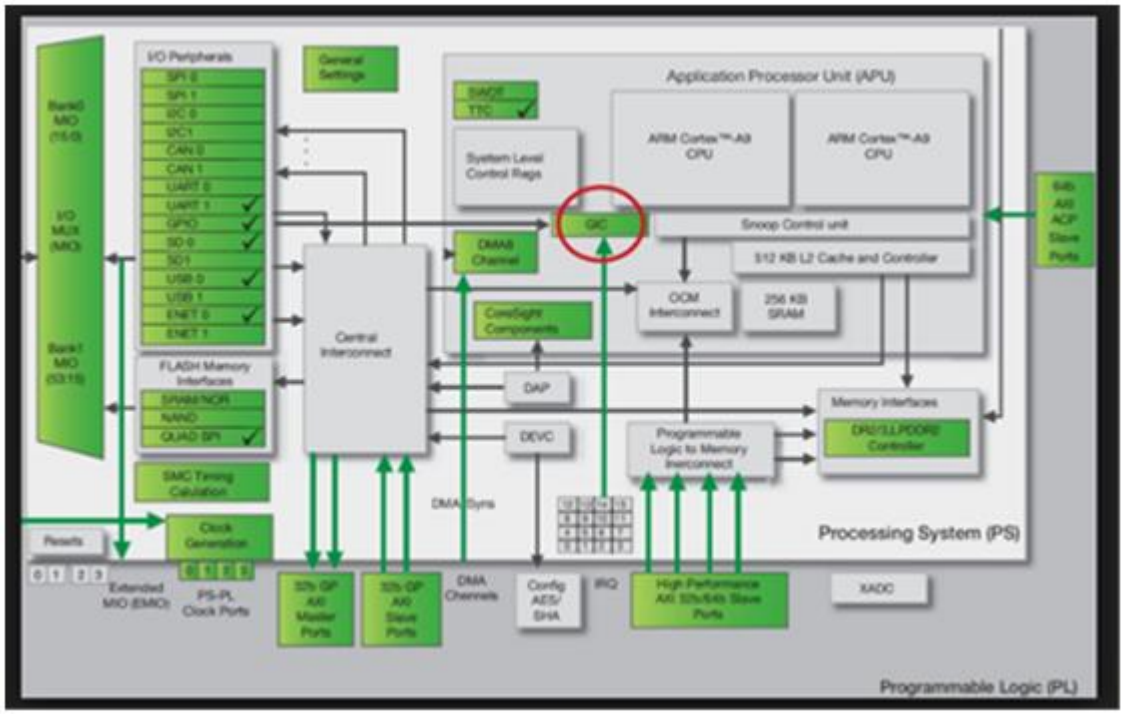

Fig.1: Zynq Architectur (Source: Xilinx Manual)

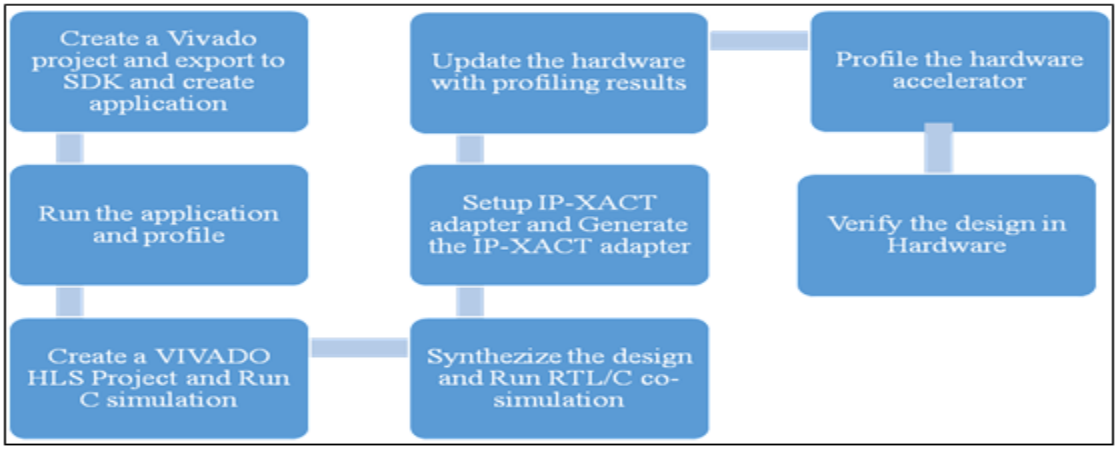

Fig.2: Profiling Flow

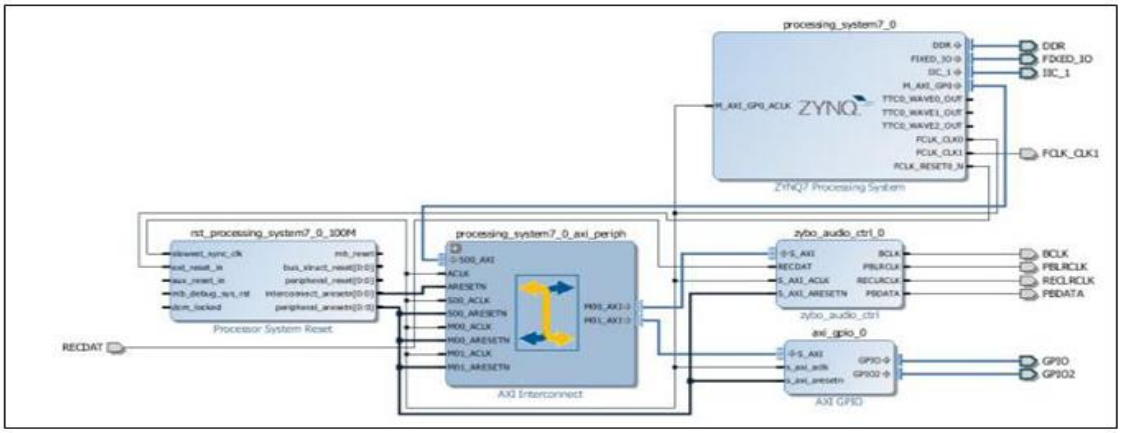

Fig.3: Block design for connections made for the Zybo Board

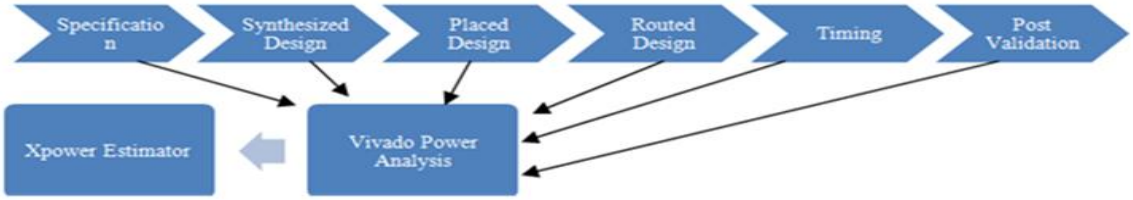

Fig.4: Vivado Power analysis Flow

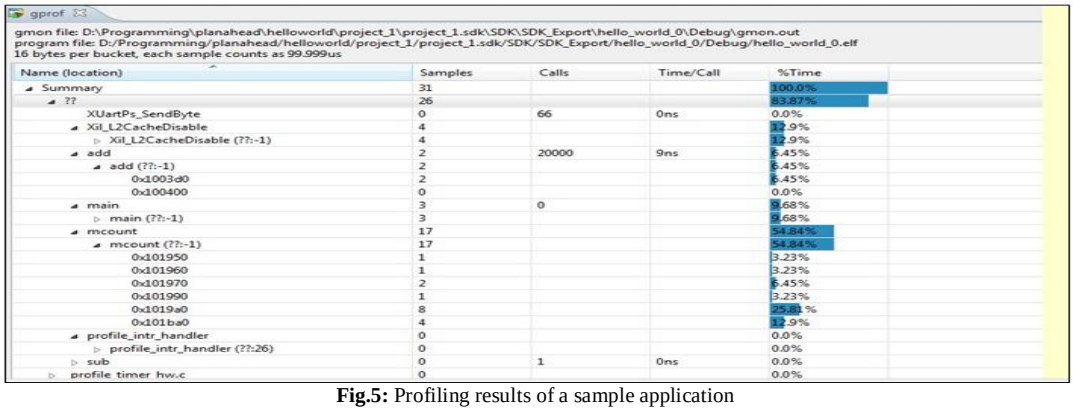

Fig.5: Profiling results of a sample application

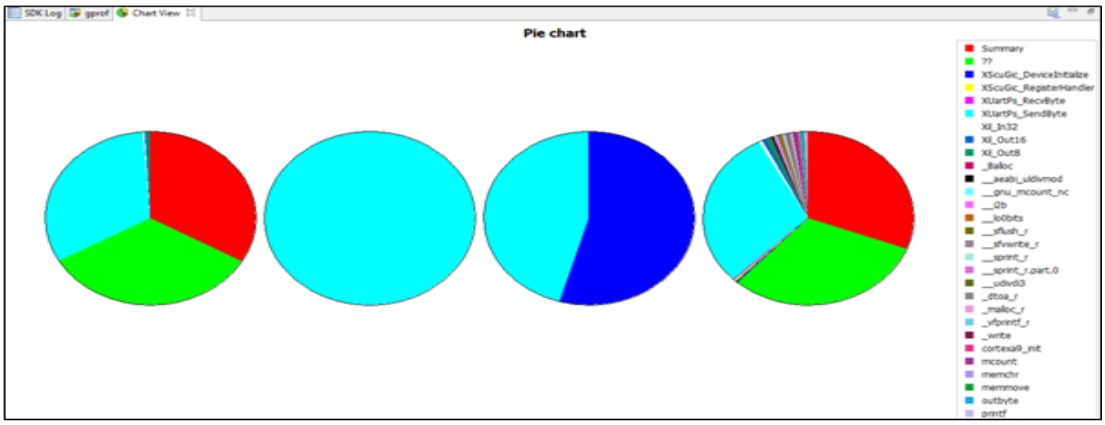

Fig.6: Pie chart representation of time consumed by each function

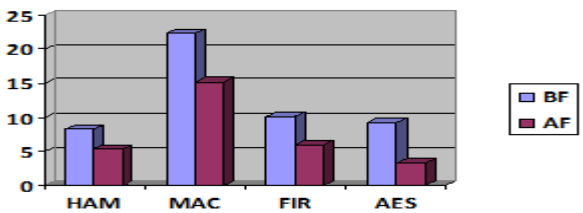

Fig. 7: Bar chart representation of power reduction after profiling