Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

ANALYSIS OF SOFTWARE POWER CONSUMPTION OF EMBEDDED DSP SOFTWARE

C.V. Krishna Reddy

Nalla Narasimha Reddy Education Society's Group of Institutions, Hyderabad.
cvkreddy2@gmail.com

Abstract

For low power application, power is a major design constraint. Its consumption during software program execution is a critical consideration for constructing low power embedded systems. This software power can be diminished in a variety of ways. Software related power could be lowered by changing the instruction of a code. When it comes to building embedded applications, power is becoming a key limitation. To predict the to estimate the present power consumption of an embedded digital signal processor, a new power analysis model was developed in this research. Power is becoming a major constraint when it comes to developing embedded applications. Power analysis methods that use circuit-level or architectural-level modeling to forecast the power consumption of a particular piece of software are either ineffective or incorrect. This paper develops an embedded DSP instruction-level power analysis model using physical current measurements. There are major discrepancies between the software power model for this unique DSP processor and previous power models for different general purpose commercial microprocessors. The circuit condition affects an instruction stream's power consumption more noticeably in this DSP processor. Dual memory access and instruction pair packing are also possible because of the processor's architecture, which includes these two characteristics. The amount of energy saved as a result of implementing these features is being studied. The processor's on-chip Booth multiplier consumes a significant amount of energy while running DSP programs. A microarchitectural power model for the multiplier is developed and evaluated for further power minimization. Novel instruction-level power model-driven scheduling approach is offered to harness all of the above impacts. To demonstrate the success of this method, several example programmes are presented. There have been energy savings ranging from 26 percent to 73 percent. These energy savings are real, as evidenced by tangible measurements. It's worth noting that the energy savings are basically free. It is obtained by modifying software and so does not require any software. Instead of the traditional analysis the power consumption of an algorithm as a whole is evaluated at the behavioral level, which is done at the instruction level. Our energy models for a Texas Instruments DSP provide some findings, but not all of them. To provide an overview of embedded DSP power consumption models and energy-saving methods at the software level, this article was written. The literature-based software level power consumption models are described, together with their benefits and drawbacks. For the processor, an improved software level power consumption model is proposed, which is shown to attain an accuracy of 96.8% or higher across a variety of benchmarks.

KEYWORDS: DPS, methodology and analysis, levels, use of DPS, advantages, Low power, high power, techniques.

Introduction:

Because of their optimum performance at minimal power, DSPs are the most suitable processors for this kind of application. Because of the rising demand for portable

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

systems and the tendency toward high-level integration and higher operating frequencies in technical terms, the importance of power restrictions during embedded system design has steadily increased in recent years. As a result, there has been a substantial amount of research into estimate of power consumption and development of low-power structures. Software programmers can use power simulators (profilers) to determine hotspot, high-initial step toward reducing the amount of power used by certain sections of their program code. Developers of power simulators must include a comprehensive model of power usage into their products. It is not possible to use the existing CPU power simulators to build higher-level components such as systems or applications. The power consumption of embedded software cannot be evaluated using these techniques until the conclusion of the design process when the application's power consumption is understood. This paper presents a method for evaluating the power consumption of a VLIW DSP while it is executing a software program. An early estimation of the core processor's power consumption during the execution of a software program is the goal of this study. Texas Instruments' TMS320C6416T (referred to as C6416T for the remainder of the article) is the DSP under consideration. In the C6000 DSP platforms, this processor has the best performance among the fixed-point DSPs. A power analyzer analyzes the voltage, current, and other electrical properties of electrical equipment in a power system. The harmonic and frequency spectra of voltage and current may also be examined with this device. The goal of this study is to investigate the theory and design of a low-power DSP-based power analyzer in more depth. The results of the simulation experiment indicate that the high precision harmonic analysis technique of our system's software can fully satisfy the accuracy criteria of the power analyzer.

Methodology and functional analysis:

A). Framework of the methodology:

The goal is to create power optimization strategies for a C algorithm using DSP programming. Then, using a toolkit called Consumption estimation, we estimate the power consumption of all the individual primitives and, by combining these estimates, we evaluate the algorithm's overall power consumption. The Optimization toolbox is used to rewrite the C algorithm.

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

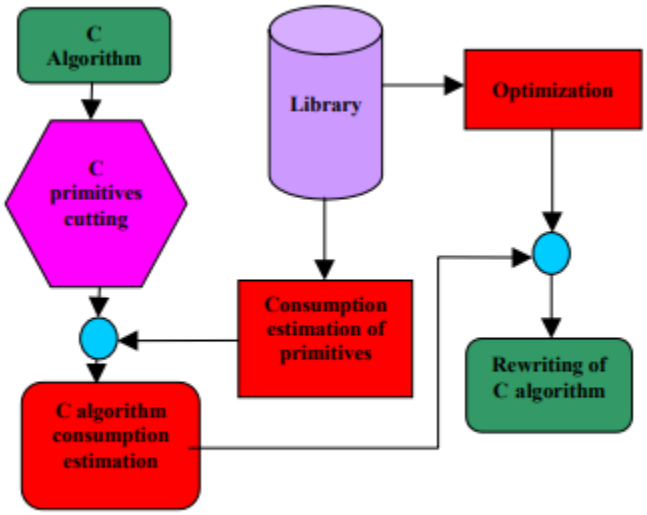

Figure 1: Methodology of power estimation and optimization

Creating a comprehensive library for both the Consumption estimating and Optimization toolkits necessitates taking exact measurements and correlating them with correct power estimations.

B). General functional analysis:

We utilise a modern DSP architecture and pipeline power consumption is used to do a functional analysis. Owing to its deep pipeline, VLIW instructions, and cache-mode-capable internal memory, the processor can have a complex design... We begin to group architectural elements based on how much energy they waste and how they interact with one another. It is possible to tailor the overall representation to a specific DSP by adding or deleting clusters. We identify the interconnections between the various clusters and classify them based on their power usage. Because it contains all the control registers, the CU interacts with every other cluster in the system. Additionally, data from an external memory is loaded through DMA into the MMU and PU to establish communication. The IMU (Instructions Management Unit) and the PU have a relationship since the instructions are loaded in the IMU and subsequently executed in the PU. Finally, the link between the DMA and fetch phases results in a final interaction between the MMU and the IMU.

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

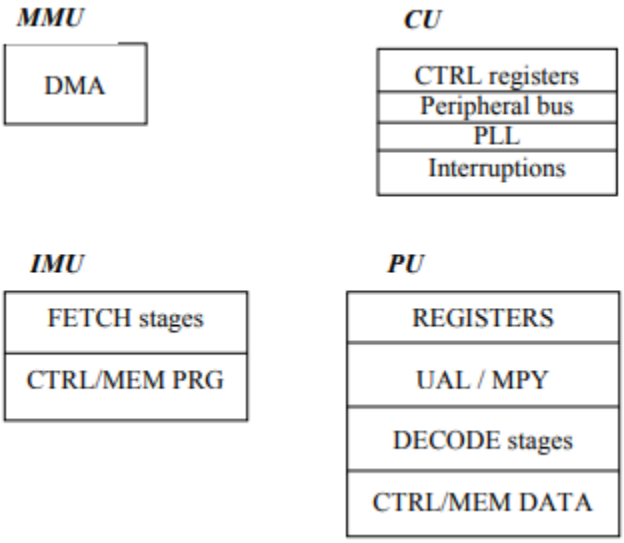

Figure 2. Functional Analysis

C). Specific functional analysis:

The functional analysis was performed on the TMS320C6201, a TEXAS INSTRUMENTS DSP from the last generation. Because it features a long internally-programmable memory that may serve as cache memory; pipe; VLIW instructions; parallelism, this processor has a complicated architecture. The External Memory InterFace (EMIF) block is utilized for all external access to the program memory (in the IMU) and the data memory in this kind of DSP (in the PU). As a result, we have decided to include it in all of the relevant sections.

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

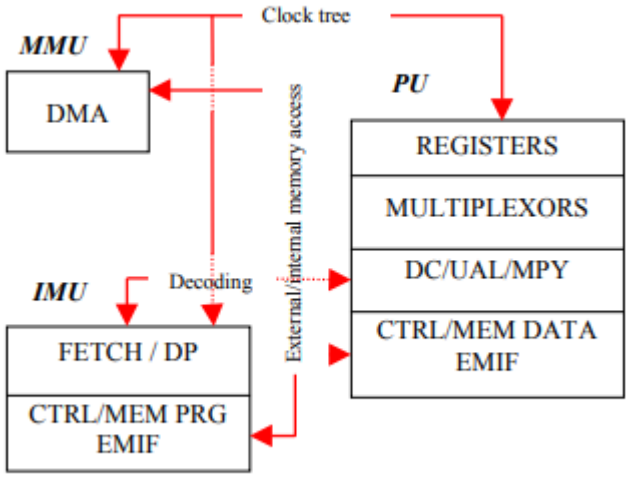

Figure 3: Interactions for the TMS320C6201

Figure 3 shows how our functional analysis approach was applied to this specific DSP. Only interactions between clusters are described here for the sake of clarity; while inter-cluster interactions have been investigated, they are not included. Using the DSP's two register files and multiplexers, the datapath to be crossed. It also emerges in the form of a new cluster. The consumption laws listed below characterise the qualifying of various interactions. These principles enable us to calculate the amount of electricity used by each part of a programme and, as a result, the entire algorithm.

Software Power Consumption:

This research, on the other hand, has largely overlooked the potential energy savings that might be realised by software optimization. This was mostly owing to a scarcity of practical approaches for assessing programme energy use. The prior sections' methods for measuring and estimating helped to correct this flaw. The growing trend toward tighter energy budgets involves the identification and examination of every available source of energy savings, prompting us to rethink software design. The energy cost of a program is calculated as the average current multiplied by the running time. Consequently, to decrease energy expenses, the product's value must be lowered.

Levels:

A. **Instruction Reordering Description:** A method to minimize the number of switches in the control route of an experimental RISC processor by scheduling instructions accordingly. This approach seeks to minimize the program's average current using the energy formula by reordering instructions. However, On the basis of real energy measurements on the 486DX2, our research reveals that

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

this approach does not result in a substantial reduction in total energy. Using this method, we hope to minimize the so-called circuit state overhead. This number, as we saw previously, is limited by a narrow range and shows minimal variation. A mere 2% difference in cost was found by rearranging numerous sequences of instructions in an alternative order. For the 486DX2, it may be stated that this strategy is ineffective.

B. **Generation of Energy Efficiency Code:** This may have little effect on overall energy expenditures, but the instructions that are utilized to create the new code can have significant effects. 486DX2 instructions, for example, have substantially greater average current when using memory operands than when using register operands, according to an analysis of their energy expenditures. Using just register operands in an instruction consumes around 300 mA. Memory reads from the cache can consume up to 430mA. Because memory operands incur more cycles, the even larger savings would be realized in energy costs, i.e. the average current multiplied by the operating time. If the cache is hit, ADD DX, I requires two cycles whereas ADD DX, BX just takes one cycle to complete the operation. All of these may add time to the process: pipeline delays, mismatched accesses, and cache errors. Reduce the number of memory operands by using suitable code generation rules, such as storing the least amount of context during function calls. The most effective technique to reduce memory operands, however, is to make greater use of registers. Techniques like efficient global register allocation of temporaries and commonly used variables come into play here.

Why Use a DSP?

A DSP is essentially a type of microprocessor. It contains the same basic properties and components as a computer, such as a CPU, memory, instruction set, buses, and so on are all components of a computer systems architecture. For the most part, nothing has changed save for the fact that some procedures have been made to run more smoothly. DSPs are designed for high-speed numeric processing and fast, real-time processing, thus they have hardware and instruction sets that reflect that of analogue data from the environment. We'll get into specifics later. The CPU, as well as the memory, instruction sets, buses, and other components, has been tweaked slightly.

A digital signal processor (DSP) a signal processor with high throughput and efficiency. DSPs consume less power since they are specialized in signal processing to complete the task, similar to how we do in life. When performing signal processing tasks, DSPs use far less effort, time, and resources than a general-purpose CPU. When it comes to specialising a processor, it's critical to focus on the sections that are used the most. It's pointless to develop anything efficient at doing things that aren't used very often! Concentrate on the areas where you can get the most bang for your dollars. It's all

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

about signal processing DSPs, on the other hand, are designed to do it right the first time. DSPs may not be as effective as other processors when it comes to non-signal processing techniques. Consequently, it is essential that you understand your application and choose the proper processor.

Software optimization is also required for digital signal processing. Even with all of a DSP's fancy hardware improvements, it still takes All of this would be impossible without some powerful tool assistance, particularly the compiler. The compiler comes in handy when attempting to translate from a language like C into object code that can be run on this specific CPU. The process of writing code that completely "entitles" Compilers have a difficult time optimizing for the DSP hardware platform because of its complexity. As with any other processor, the DSP's architecture must be such that it can react rapidly to real-world events, transfer data quickly in and out, and perform speedy analyses on data. Previously, we spoke about how this would be processed. Real-time applications often have a barrier in the data input and output speeds rather than in processing speed. Digital signal processors (DSPs) are designed to fulfill this need in the real world. DSPs are equipped with a variety of peripherals, including high-speed I/O ports and buffered serial connections. To accommodate this. Because of the rapidity with which they can handle streams of data, DSPs are frequently referred to as data pumps. This is also another feature that distinguishes DSPs.

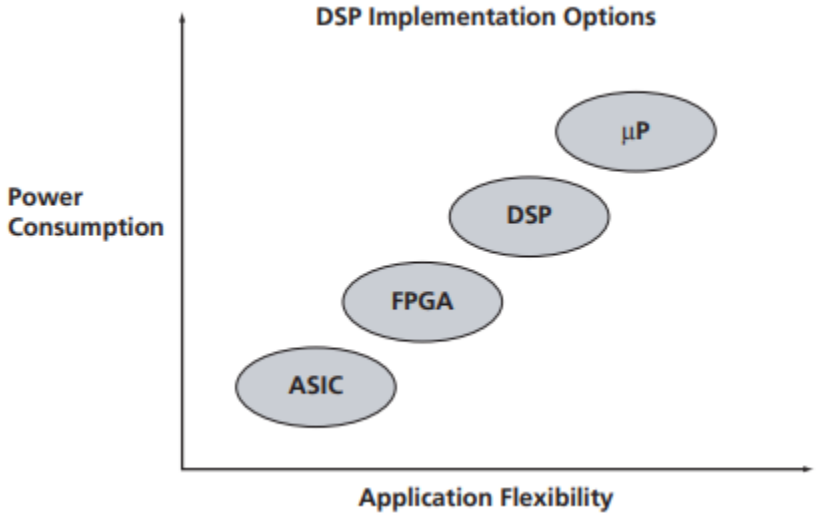

Figure 4. DPS Implementation Options

In deciding between programmable and fixed functions, two key tradeoffs must be made: flexibility and power. DSP's use of the word "digital" necessitates discrete signal processing in order to portray data as readily manipulable numbers. To put it another

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

way, the signal is quantitatively represented. One or more signal characteristics, such as time, have been quantized in this kind of representation.

Advantages of DSP:

The use of a digital signal processing system has many benefits over an analog one. These are only a few examples:

• **Changeability** – Digital systems may be easily reprogrammed for new purposes or fine-tuned for current ones. DSPs enable for quick and simple application updates and modifications.

• **Repeatability** – The properties of analog components may vary somewhat as a result of variations in temperature or time. The repeatability of a programmable digital solution is greatly enhanced by the system's programmability. For example, several DSPs in a system may execute the exact same program again. Analog signal processing necessitates custom tuning for each digital signal processor (DSP) in the system.

• **Size, weight, and power** – Utilizing mainly software components in a DSP solution uses less power overall than using just hardware components.

• **Reliability** – Analog systems are only as dependable as their hardware components. If any one of these components breaks down because of a malfunction caused by a physical condition, the whole system will deteriorate or stop working. Insofar as the program is implemented appropriately, a DSP solution will work successfully.

• **Expandability** – The architect should include additional equipment in the framework to make it more functional. Maybe it is not possible. Installing comparable functionality to a DSP is as easy as adding software.

Applications for DSP Software:

Low Power DSP software:

Low-cost alternatives such as DSPs are becoming more popular in a variety of applications. Electronic motor control is a popular topic. Electric motors can be found in a wide range of consumer devices, including washing machines and refrigerators. The appliance's electric motor uses a significant portion of the total quantity of energy. Changing the motor speed has a significant effect on how much energy an appliance uses in total. Modern three-phase variable-speed drive systems are used by manufacturers to obtain the desired performance increases required to meet the appliances' energy consumption specifications. In order to create increasingly sophisticated motor drive systems for various household appliance applications, DSP-based motor control systems provide the required bandwidth. Three-phase induction motor speed control variable is provided by a low-cost DSP, which improves system efficiency.

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

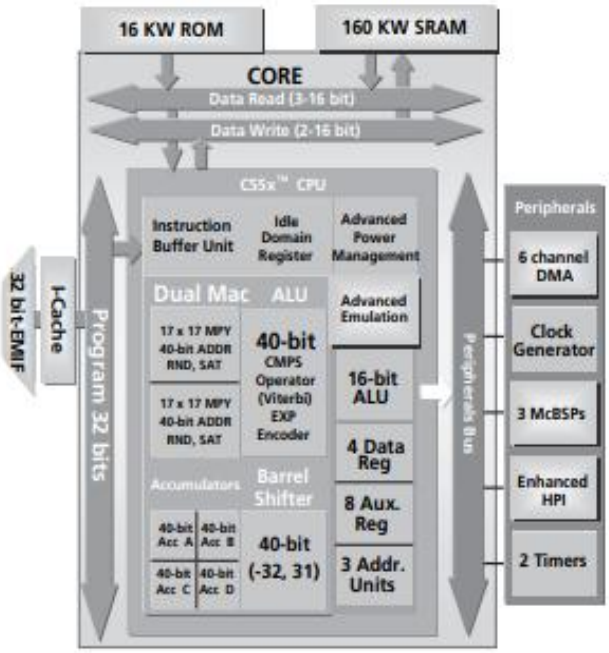

Fig 5. Low Power DPS software

DSPs to control variable speed blowers that significantly further develop energy proficiency. Minimal expense Variable speed controls, which do not need mechanical equipment, are made possible by DSPs, which are widely used in clothes washers. Sensorless control is provided by DSPs for these devices, eliminating the need for speed and current sensors. Improved detection and management of off balance enable faster twist rates, which dry clothes more quickly with less noise and vibration.

Power Efficient DSP Software:

The batteries that power these systems are essential. The greater the battery life extension, the better. As a result, it's only natural require CPU power to be taken into consideration by the systems' designers. Longer battery life and the ability to run these systems and applications are made feasible by using a CPU that consumes less power. Systems dissipate less heat as a result of lower power usage. As a result, expensive hardware components like as heat sinks are no longer required to adequately dissipate heat. Because there are fewer components, the total system cost is lower, and the overall system size is smaller.

These systems require high power efficiency not just because to power supply limits, but also due to heat dissipation issues. The boards in these systems are extremely dense, having a significant number of components per board. In a small space, there could be numerous boards per system. Reduced power consumption and heat dissipation are important to the designers of these systems. Increased performance and

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

density can be achieved using low-power DSPs. Using fewer heat sinks and cooling systems results in systems that are less expensive to develop and manufacture.

High Power DSP Software:

DSPs high-speed signal processing is performed using novel designs at the performance's extremes. To attain great speed, advanced architectures like the very long instruction word (VLIW) employ a lot of parallelism and pipelining. To attain this performance, modern architectures make use of other technologies such as compiler optimization.

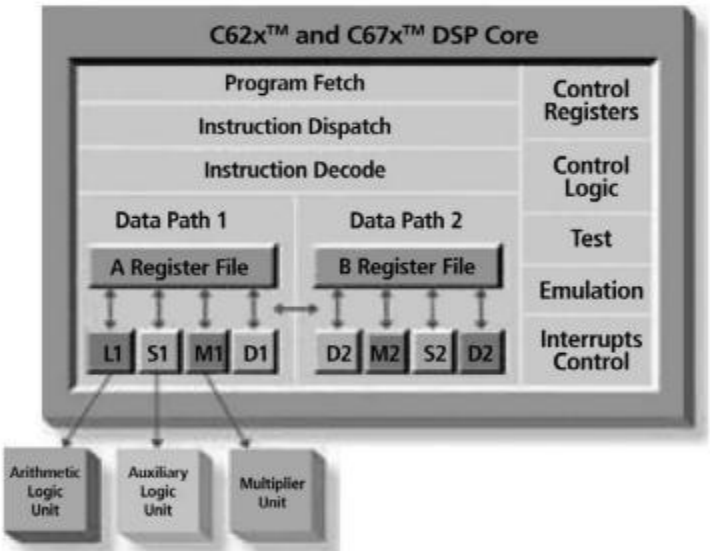

Fig 6. High Power DPS Software

Power consumption Techniques in DSP Software:

Much of the reduction in DSP power usage is due to improvements in process technology. Reduced operating voltages are required for smaller-geometry transistors in the logic of the core. The engineer must study the application during the DSP application life cycle to find the proper balance between active and standby power usage. This is a critical design consideration that influences the DSP selection. Active power consumption is the main force if you are dealing with a big, heat-sensitive system.

DSP software has fused various power saving and enhancement procedures into the models of these gadgets. Beside the normal investment funds coming from proceeded with progress in semiconductor processor innovation, design propels have likewise been made to current DSP gadgets. These incorporate:

• Expanded for the parallelism and dedicated directions for a class of DSP activities, which enables the program to wait until a hinder or reset occurs. Includes the inactive

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

direction When a device is not in use, power saving modes depend on a setup register to save power.

- Broad utilization of the creation of a clock tree and the reason for using it.
- Using internal buffer queues as a way to reduce the amount of memory accessed.
- Address and data routes that are uniquely created for each client.
- The method was created with little leakage.

The following steps are required to calculate overall device power consumption:

1. Calculation apportioning – The calculation viable ought to be broken into areas of special gadget action and the power necessities for these segments ought to be determined independently. These areas would then be able to be time arrived at the midpoint of to decide the general gadget current prerequisites.

2. Central processor movement – The current commitment to computer chip action can be controlled by inspecting code and deciding the time-found the middle value of current for every calculation parcel.

3. Memory use – Scale the current in accordance with the amount of memory being used, up or down by 2. Using on-chip memory saves on power since it uses less power. In addition, switching to ROM instead of Smash is less effective.

4. Peripherals – The additional current required by the clock, sequential port standards, and interaction with ports should all be taken into account.

5. Current because of yields – Consider the current needed by the calculation to work the outer location and information transports.

6. Estimation of normal current – Assuming the power supply is fully used span of gadget movement, various portions of action will display distinctive current levels for various time allotments.

7. Temperature and power supply voltage effects on device operating current – Incorporate which components' consequences have been identified after a complete device current.

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

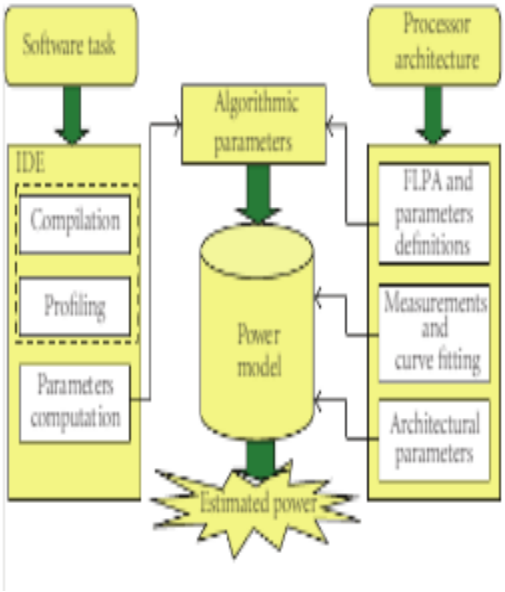

Fig 7. Power model

One or more performance indices, such as latency or power consumption, are maximized or minimized in embedded applications:

- Throughput (or execution speed)
- Memory usage
- I/O bandwidth and
- Power dissipation

Conclusion:

We've developed a high-level power estimation method that uses a DSP architecture behavioral analysis to estimate the energy of a whole algorithm rather than the traditional methodology of estimating energy at the instruction level. This approach was implemented on a Texas Instruments DSP (TMS320C6201) from the latest generation, which has a complex architecture. In assembly, a FIR algorithm's results are checked by measurements with an error of less than 8%. The project's eventual goal will be to expand this promising method to the estimation of C language algorithms in future studies, after it has been enhanced for other applications.

A way for assessing embedded software's energy usage. It's built on a model at the instruction level that estimates the energy costs of individual instructions as well as the implications of communication between instructions. Three factors led to the development of the analytical technique. It shows how much energy is being used by

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

the CPU. An embedded design may be verified using this tool adheres to its energy limitations, as well as to drive the development of embedded software that adheres to certain restrictions are in place. There has been considerable power reduction in first efforts at code re-writing, supporting the need for such a power analysis method A CISC and a RISC have been used to test the method so far. Future research will compare and contrast energy consumption models of various architectural styles. DSPs, superscalar processors, and processors with integrated power management will all be scrutinized. Finally, we aim to use the findings of this study to develop automated methods for reducing embedded software's power usage.

To assess the power consumption of a programmable processor running embedded software, we fostered an exact practical level assessment strategy. The targeted platform is Texas Instruments' VLIW DSP C6416T, which is available commercially. In our proposed model, the inter-instruction and pipeline slow down impacts have been explored. The legitimacy and accuracy of our model were shown by computing the power consumption of an assortment of normal sign and picture handling strategies. We likewise tried the accuracy of our constructed model on a true application in the field of video handling. In spite of the fact that our strategy for assessing In comparison to other VLIW processors, the TI C6416power T's consumption is significant, we may return to the division of the processor engineering into utilitarian squares for various designated processor plan.

Digital signal processors (DSPs) are useful in embedded systems for a single reason: signal processing. Embedded processing cannot match DSP's ability to execute complicated sign preparation functions in real time. These computerized signals should be treated with respect additional preparation by DSPs, which should respond constantly to simple climate indications and convert them back to basic structure for re-sending to the climate when necessary.

As DSPs and embedded applications are getting more complex, and the process of integrating hardware and software is the most time-consuming part of the development process. Support for integration tools has emerged as one of the most crucial factors in reducing time to market for complicated embedded DSP projects. When it comes to the integration and testing of complicated applications, DSP emulation and debugging can be a huge help.

Xi'an Shiyou Daxue Xuebao (Ziran Kexue Ban)/
Journal of Xi'an Shiyou University, Natural Sciences Edition
ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

References:

1. M. E. A. Ibrahim, M. Rupp, and S. E.-D. Habib, “Performance and power consumption trade-offs for a VLIW DSP,” in Proceedings of the IEEE International Symposium on Signals, Circuits and systems (ISSCS ’09), pp. 179–200, IEEE, Iasi, Romania, July 2010.
2. S.M. Deckmann, J.A. Pomilio Avaliação da Qualidade da Energia Elétrica Faculdade de Engenharia Elétrica e de Computação – Unicamp (2011) (in Portuguese). Retrieved November, 2014.
3. D.D. Ferreira Análise de Distúrbios Elétricos em Sistemas de Potência. 2010 Universidade Federal do Rio de Janeiro (2010) Doctoral Thesis (in Portuguese). Programa de Pós-Graduação em Engenharia Elétrica (COPPE).
4. Ghaderi, M.P. Moghaddam, M.K. Sheikh-El-Eslam Energy efficiency resource modeling in generation expansion planning Energy, 68 (2014).
5. R. Hackl, S. Harvey Applying exergy and total site analysis for targeting refrigeration shaft power in industrial clusters Energy, 55 (2013)
6. M. Misiti, Y. Misiti, G. Oppenheim, J. Poggi Wavelet toolbox for use with MATLAB The MathWorks, Inc. (2012).
7. V. Musolino, A. Pievatolo, E. Tironi A statistical approach to electrical storage sizing with application to the recovery of braking energy
   Energy, 36 (11) (2011),
8. L. Peng, Y. Zhang, Y. Wang, X. Zeng, N. Peng, A. Yu Energy efficiency and influencing factor analysis in the overall Chinese textile industry
   Energy, 93 (2015).
9. D. Wang A dynamic optimization on economic energy efficiency in development: a numerical case of China Energy, 66 (2014).
10. Zhongru Shen, Hui Ding, "intelligent instrument design", Department of Measurement Technology and Instruments, School of Electrical Engineering, Xi'an JiaoTong University, 2012.
11. Rong-Ching Wu, En-Chih Chan, Jia-Chu Lee, Chun-Wei Huang. Implement of DSP Based Optimal Power Analyzer, IEEE 2nd International Symposium on Next-Generation Electronics (ISNE)-February 25-26, Kaohsiung, Taiwan, pp. 153-15, 2013.
12. Xie Jun and Zhu Lei, "Design of a Multi-Port Data Collector[J]", *Information Communication*, no. 3, pp. 95-96, 2016.
13. Yi Longjiang, Gao Junwei, Zhang Zhiqiang et al., "Design of Rail Transit Data Acquisition System Based on LabVIEW[J]", *Journal of Qingdao University (Engineering & Technology Edition)*, vol. 31, no. 3, pp. 44-48, 2016.
14. Zhou Dan, "rail transit station design energy collection and energy-saving control system", *[J] electronic world*, no. 13, pp. 82-83, 2015.
15. Yang Linfang and Zeng Baoquan, "Design and Implementation of Energy Consumption Data Collector Based on MQX [J]", *modem computer*, no. 21, pp. 69-73, 2017.
16. Wang Wenqing and Yao Wei, "Design of data collecting device of energy consumption based on ARM and WiFi*[J]", *Sensors & Microsystems*, vol. 35, no. 8, pp. 115-118, 2016.

ISSN:1673-064X
E-Publication: Online Open Access
Vol: 64 Issue 11 | 2021
DOI 10.17605/OSF.IO/8FKQS

17. Han Xiaona, Chen Chaoxu and Yu Linfeng, "Active fault to lerant control for sensor failure[J]", *Journal of Northwest University (Natural Science Edition)*, vol. 40, no. 1, pp. 39-42, 2010.
18. Mao Haijie, Li Wei and Feng Xiaolin, "Review of active fault tolerant control for nonlinear system[J]", *Transducer and Microsystem*, vol. 33, no. 4, pp. 6-9, 2014.
19. .Huang Rimao, Qiu Xuesong, Gao Zhipeng et al., "Fault Detection Method for Neighbor Data Analysis in Wireless Sensor Networks[J]", *Journal of Beijing University of Posts and Telecommunications*, vol. 34, no. 3, pp. 31-34, 2011.
20. ZHANG Shao-jie, LIU Chun-sheng and HU Shou-song, "Adaptive Fault-tolerant Control of Actuator with Multiple-input and Multiple-output Minimum Phase System [J]", *Control Theory & Applications*, vol. 27, no. 9, pp. 1190-1194, 2010.