International Journal of Emerging Science and Engineering (IJESE)
ISSN: 2319–6378, Volume-3 Issue-8, June 2015

Power Consumption in the Embedded System

Saud Salem Almusallam

Abstract— In the embedded system the Processors are the computing elements. Most of the embedded systems are portable battery powered systems. The system depends on the battery life, which is determined by the power consumption of the entire system. So This paper Shows and presents -from starting of the embedded system definition- the factors of power consumption of an embedded system Processor.

Index Terms -- Introduction, Embedded systems characteristics, Low Power Design, Average Power Optimization, Memory System, FPGA, Power Reduction Techniques, Conclusion, References

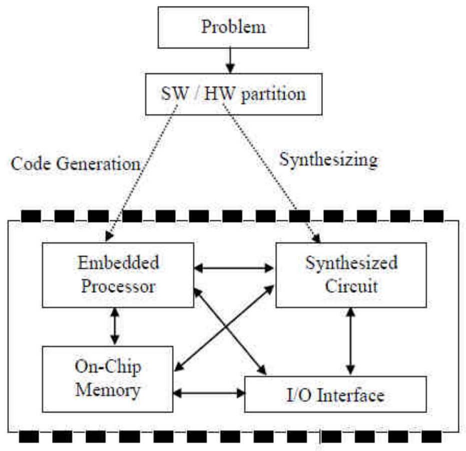
I. INTRODUCTION

The embedded system: is some combination of computer hardware and software, either fixed in capability or programmable, that is specifically designed for a particular function. Industrial machines, automobiles, medical equipment, cameras, household appliances, airplanes, vending machines and toys (as well as the more obvious cellular phone and PDA) are among the myriad possible hosts of an embedded system. Embedded systems that are programmable are provided with programming interfaces, and embedded systems programming is a specialized occupation. Processors of different size and model are embedded into devices for a variety of reasons. Perform computations, provide automatic control, development of applications like embedded, biomedical, multimedia and control, provide communication and control over internet and many more. Processors are the brain of these systems. Since the scope of the processors vary widely, their computation power also changes from few hundred instructions to millions of instructions per second. And this makes it hard to compare different processors and evaluate their performance. The existing competition between different manufactures of processor makes it very competitive in terms of increased performance, reduced size, increased clock frequency etc. which puts a tradeoff between the performance and the power consumption. The consumer demands namely compactness in size with all sophistications added up, need of prolonged battery life etc. are hard to attain. With the dvancements in technology, the transistor count on a single CPU has crossed 2.5 billion transistors. Integrating these transistors for performance enhancement will also have an impact on the power consumption, because adding more and more transistors give rise to increase power consumption, thereby reducing the battery life due to the increase in the heat dissipated in the device; which reduces the usefulness of the portable embedded system. Due to this, power management has become a design constraint today, for most of the computationally intensive and sophisticated applications.

Revised Version Manuscript Received on June 23, 2015.

Saud Salem Almusallam, Alshaab B2 ST 24, Kuwait, Kuwait City 00000, Kuwait.

In order to improve the utility of the embedded system, the power consumption of the entire system has to be minimized. Since the major computations are done by the embedded rocessor/controller, energy minimization of the processor is also vital for the total power reduction, even though it is relatively small. The Intel corp. has cancelled one of its new generation processor Tejas Pentium 4 processors due to the power related issues. But, for large and complex applications like smart grid, sensor network etc, where thousands and lakhs of processors are present and are performing computations for almost all the time, the cumulative energy consumption of the entire nodes in the network cannot be neglected. This paper focuses on the energy consumption issues on a processor level rather than on the system level. There are available plenty of techniques for power consumption reduction on a system level and relatively there is only little work which aims at power reduction of the embedded processor. The main objective of this paper is to identify the different factors that are affecting the execution of the processor and yielding to power consumption. The Embedded systems are presented in most of the electronic devices and instruments used in daily life; they can be found in consumer electronic devices (calculator, digital cameras, cell phones, etc.), office equipment (printers, copy machines, fax machines, etc.), home appliances (microwave ovens, washing machines, alarms, etc.), and automobiles (cruise control, transmission control, fuel injection, etc.). The embedded system is any computing system other than a desktop computer. An embedded system typically consists of four main components: an embedded processor, synthesized circuit for dedicated hardware units, memory, and I/O interface. All of these are typically implemented in one chip constituting what is called a *systemon-chip* (SOC) as shown in Figure



Embedded system structure

II. EMBEDDED SYSTEMS CHARACTERISTICS

- **Single function:**

An embedded system usually executes a certain task (or program) repeatedly.

- **Real-time operation:**

Time constraint is very crucial in execution of tasks on embedded systems. Even a small execution delay might cause a serious malfunctioning or total failure.

- **Tight constraints:**

Because of the nature of embedded systems, their design metrics such as size, performance and power impose tight constraints. Embedded systems are specified in different levels of abstractions. Gajski's famous Y-chart has identified different views of these abstraction levels and the relationship among them. Typical levels of abstraction are the system, the behavioral, the register-transfer (RT), the logic, and the physical level. Each level of specification is a refinement of the level above it. At the system level, the design is described as a set of interacting subsystems (processes) to be mapped to either hardware or software components. These subsystems can be implemented using processors (software), application-specific IC's (ASICs), memories and dedicated hardware. At the behavioral level, each subsystem is specified in its algorithmic (or functional) form. At the RT level, the system is specified as a collection of communicating register transfer logic (RTL) units such as ALUs, registers, and multiplexers. The logic level specification is the hardware implementation of the logic functions given as a netlist of logic gates and flipflops. The physical level description is the physical implementation given as a netlist of transistors, capacitors, and resistors on a board. Register-transfer (RT), logic, and physical levels belong to the hardware side. Module level and block level specifications are the typical levels of abstraction on the software side.

Low-Power Design

The usual design metrics of embedded systems are performance, size, testability, and power. Although conventional design metrics such as performance, size and testability are important, the most critical design metric nowadays is power. The demand for long-life batteries within tolerable size and weight and the reliability of integrated circuits are the main factors that dictate power-aware design of embedded systems. Reliability of integrated circuits is tightly related to the peak and average power consumption. It has been estimated that every 10 Degree C increase in operating temperature causes component failure rate to approximately double. Since most of the power consumed by the integrated circuits is dissipated in the form of heat, complex cooling techniques are needed to maintain the operating temperature of the circuits within the normal level. Moreover, heat dissipation is a limiting factor in increasing the number of transistors in a chip. These and other factors such as cost and size have driven the low power design to be a critical issue. Using low power design techniques helps in solving these problems and alleviates chip failure and system operation degradation. The main sources of power dissipation in CMOS circuits are the capacitive switching power, Psw, the short-circuit power, Psc, and the power consumption due to leakage current, P leakage. These power consumption sources are summarized in the following equation:

Pav = Psw + Psc + Pleakage.

The capacitive switching power together with the short-circuit power is called *dynamic power*, and it is due to charging and discharging in CMOS gate. Dynamic power is considered to be a significant part in the total power consumption and is given by the following equation

$$P_{dynamic} = \frac{1}{2}\, \alpha\, C_L\, V_{dd}^2\, f_{clock},$$

where *CL* is the load capacitance at the gate output, *fclock* is the circuit clock frequency, *Vdd* is the supply voltage, and *α* is the average number of transitions per clock cycle at the gate output, referred to as the *switching activity*. Power/energy reduction in embedded system can be achieved by carefully designing each of its constituent components targeting low power/energy design. In dedicated hardware units, power/energy reduction is usually achieved through optimizing dynamic power consumption in the main high-level synthesis tasks, namely scheduling, allocation, and binding. In processor cores, dynamic power-management techniques, compilation-based techniques such as code transformations, adoption of domain-specific instructions and specialized addressing modes, and dynamic voltage scaling are used extensively to achieve power/energy reduction within performance constraint. In memory system, the central idea of temporal locality is exploited to obtain power/energy reduction by reducing memory requirements and data-transfer traffic to and from memory.

Memory System

Memory is an integral part of embedded system and it is the most dominant component in its size. Thus minimizing memory requirement helps in reducing the size and so the cost of an embedded system. In addition, minimizing the amount of memory has an impact on reducing the amount of data traffic between the SOC and the off-chip storage that reflects in power minimization. Memory optimization is a prevalent goal of the most compilation techniques in addition to performance optimization especially for the special-purpose processors used in embedded systems. In the scientific computations field, a fragment of the application program is repeatedly executed accessing large data arrays. Most of the compilation techniques use loop transformations for both performance optimization and memory minimization. Fusion is a loop restructuring technique for loop optimization in which the producer loop nest of an array is merged with its consumer loop nest inside a common loop. By fusing a common loop between producer loop and consumer loop nest, the required storage of the intermediate array is reduced by the range of the fused loop.

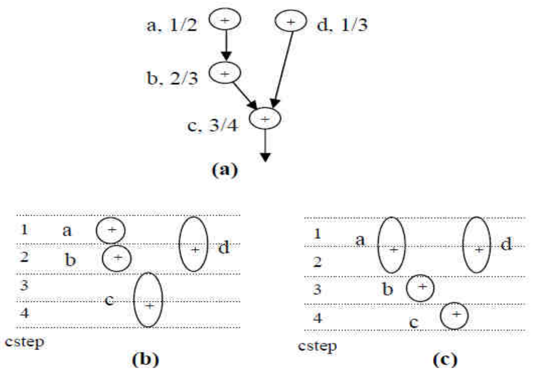


Figure: Effect of peak power consideration: (a) the DFG annotated with ASAP/ALAP (b) average power minimization alone, (c) simultaneous peak and average power minimization.

For data-dependent program regions, there are many different orders of evaluation that vary widely in the maximum memory usage required when dynamic memory allocation is used. Note that with dynamic memory allocation, a data object is allocated memory when it is needed (i.e., memory is allocated at the start of the computation that creates the data values and is used until all its parents are evaluated) and then it is de-allocated after its last use. The method that finds the order of evaluation for the least memory requirement is called the "memory evaluation order" technique. This technique can also be applied for loop optimization on the fused loops with careful consideration of the allocation and de-allocation of arrays inside some fused loop structure.

High-Level Synthesis:

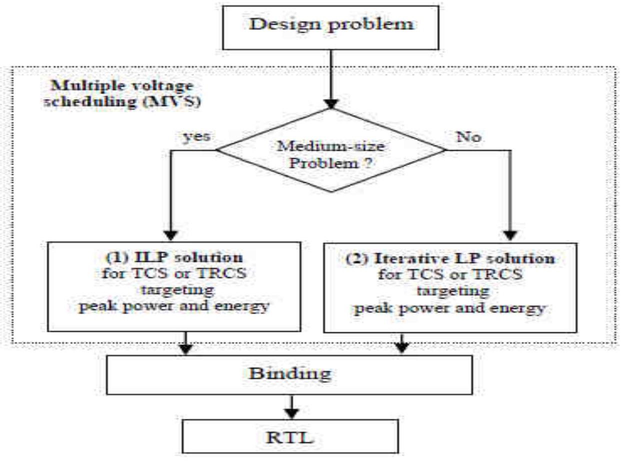
Since dynamic power (capacitive switching power and short circuit power) is considered to be a significant source of power consumption in CMOS circuits, most of the research work has focused on minimizing dynamic power consumption in HLS.

III. AVERAGE POWER OPTIMIZATION USING MULTIPLE SUPPLY-VOLTAGES

High-level synthesis (HLS) is the process of mapping the behavioral specification of the system into register transfer description. The outcome of the high-level synthesis is a structural view of the data path and a logical view of the control unit. High-level synthesis involves three main tasks: scheduling, allocation, and binding. The central task is scheduling, which is the process of determining at which control step(s) each operation in the DFG executes. We define *Scheduling for Low Power and Energy (SLoPE)* in high-level synthesis as the process of determining at which control step(s), and at what voltage level each operation in the DFG executes with the goal of minimizing power and energy. The supply voltage has a quadratic relationship to the dynamic power consumption. Thus, the most effective approach for minimizing power/energy consumption in circuits is to lower the supply voltage. However, lowering the supply voltage increases the circuit delay. Parallelization and pipelining can compensate for the increased delay but at the cost of increased area overhead. Using multiple supply voltages in the DFG solves the problem. It minimizes the power/energy consumption under latency constraint without resorting to area penalty. The idea is to assign the highest voltage level to the operations on the critical path in order to meet the time constraint and to use a lower voltage level for the operations not on the critical paths to achieve the power/energy minimization.

Problem Definition

The input to the problem include a DFG representation of the design problem, $G(V, E)$ in which each vertex $v \in V$ represents a computational operation and each edge (u,v) means that operation u has to finish its execution before operation v starts, a set of voltage levels for the operating resources, and a power/delay table that contains the average power consumption and the delay time needed for each resource operating on each voltage level and the time constraint, λ. The task at hand is to get a schedule (in which each operation is stamped to a control step, cstep $\in (1, 2, \dots, \lambda)$ and a voltage level from the set of input voltage levels) that minimizes the peak power consumption as well as the average power and energy consumption according to one of the set of constraints such as time constrained scheduling (TCS), and time and resource constrained scheduling (TRCS). We propose two solutions for the multiple supply-voltages scheduling (MVS) problem targeting peak power consumption as well as other design factors such as average power and energy consumption, and area as shown in Figure. (1) is an exact solution based on a mixed integer linear programming (MILP) formulation. (2) is a two-phase heuristic that first obtains a guided iterative relaxed LP solution of the MILP formulation followed by a *power-resourcessaving* procedure, which is a revisit of the output schedule from the first phase in which it tries to minimize the power and/or the operating resources more through scheduling operations in a lower voltage level if possible and/or through moving the operations within their new timeframes if possible without violating the peak power obtained from the first phase.


Design problem
Multiple voltage scheduling (MVS)
Medium-size Problem ?
yes
No
(1) ILP solution for TCS or TRCS targeting peak power and energy
(2) Iterative LP solution for TCS or TRCS targeting peak power and energy
Binding
RTL

Flow-chart for the DSE using multiple voltage scheduling.

IV. LOW POWER PROCESSORS NECESSITY

The power management and optimization issues were not very demanding and considered as a crucial aspect in the domain of analog circuit design. But, now it is emerged as a prevailing factor in the digital design community. At present, with the ever growing market, there is a steady and increasing requirement for low power electronic devices.

V. FACTORS CAUSING POWER CONSUMPTION

Order to incorporate the power issues in the design, many other significant factors associated with the system are to be looked into viz. the delay associated with the system, Quality of Service (QoS), performance, functional and temporal requirements of the system, throughput, reliability, area, cost etc. So, before advocating a particular low power design, these major criteria has to be looked into, to find a better system design meeting all the specifications after examining each design alternative. The following section briefs the need for low power processors. In the past, the need for low power devices were not very demanding as the clock frequency and the device density were not as high as present time, and was enough to meet the requirements. Because of this, power issues were not a major concern during that time. But today, because of humans crave for more sophistication and entertainment, most of the devices are packed with thousands and lakhs of processors in a single chip for increased performance with reduced size. This indirectly aids to the power consumption with a steady growth in clock frequency. This is supported by Moore's law which states that the transistor density doubles every 18 months. Therefore, high performance computing adds to increased power dissipation in the system. Building automation systems and other associated devices along with the computers, which are the largest and rapidly increasing electricity load, introduces an additional need of low power processors. Another factor is the escalating requirement of portable battery operated mobile devices in the market. The duration of functioning of a system depends on the battery life which is affected by the power dissipation in the system. Even though there is an appreciable growth in the device density, the battery technology has not undergone a similar growth which always puts an ever growing gap called as the *battery gap* [3]. This factor also forces the need of power consumption reduction in embedded processors. Though more compact, less weight and sturdy design is demanded by the consumers, it makes it hard to achieve these requirements without an increase in power consumption complexity of the system. As the system becomes more complex, it affects the cooling and the packing cost. The power dissipation of high computing processors is in the order of Watts (W) and the average current consumption in the range of Ampers (A) and the transient current will be a few fold of the average current which puts difficulty in the design of power supply rails. Also, unsusceptibility to digital noise is a problem to be addressed. These factors demand the need of low power processors.

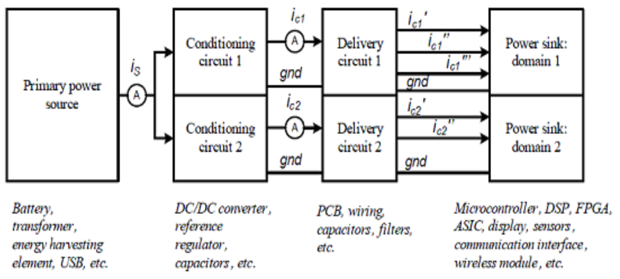
VI. FPGA AND EMBEDDED SYSTEM

Estimation of embedded systems (ES) power consumption is becoming a standard step in the development cycle of battery powered equipment. The importance of the power consumption is well realized by the vendors of integrated circuits, especially microprocessors and filed programmable gate arrays (FPGA). The issue is also targeted by the designers of ES development tools. It is also covered in the research publications. Power consumption and therefore its estimation is a key issue in designing battery powered devices in the field of wireless, medical, utility metering, etc., applications. The less power is consumed by the electronics the longer it will continue functioning without recharging or changing the battery. This is not only improves convenience of device use but is also important from the ecology point of view. Development of ultra-low power electronics opened a possibility to create absolutely new class of ES scavenging a necessary energy for their operation from the ambience. In the larger energy sector the key issue is the energy dissipation of the integrated circuits that leads to the need of cooling equipment and expensive packages. The key component of today's ES is microcontroller and/or FPGA. Both are user programmable units and their power consumption depends upon the executed program and data processed. Therefore, so called software related power is widely considered. Having in mind that most of modern microcontrollers and FPGAs are designed for run time switching between active and low power modes, the software related power is the part of overall power consumption budget that can be optimized during development of a system. To be able to minimize power consumption methods and tools for its estimation are needed. There exist two main approaches towards estimation of the power consumption of ES containing programmable unit: simulation based and measurement based. The simulation based approach uses models relating power consumption and programming instructions. This type of tools are becoming a standard utility in the FPGA design packages like Altera PowerPlay or Synopsys Power Compiler. The second approach is based on physical measurements of power consumption while running the system under development. While the measurement based approach requires measurement instruments and in most of the cases special setups inside the ES, it does not suffer from the inadequacy or even absence of reliable models. To add more, measurement approach is the only way to verify correctness of simulation based approach. Therefore measurement is important to validate the power consumption models. The goal of this paper is to give an overview of the current status of ESs power consumption measurement problems, methods and instrumentation.

Generalized power delivery model

Let us consider the generalized model of power delivery in the ES (see Fig. 1). The primary power source (battery, transformer, energy harvesting element, USB port, etc.) usually has one electric output to supply electric current and one electric contact for return current (called ground). In plenty of ESs, especially based on FPGA or digital signal processor (DSP), several voltage domains are often present. As a rule today the core of DSP or FPGA is powered by 1.2−1.8 V source while peripherals are powered by 2.7−3.3 V source. In addition, a common recommendation is followed to have separate digital and analog parts supply that can be of the same or different voltage according to manufacturer's specification. Therefore, the primary power source is followed by the conditioning circuits like DC/DC converters, voltage regulators that output appropriate voltage levels. Delivery circuits composed of PCB tracks, coupling capacitors and supply filters ensure electric source routing and shaping from the output of the conditioning circuit to the points of sinking (consumption). Power sink points are supply inputs of microcontroller or FPGA chip. It is absolutely common that a chip has several or even dozen of power supply pins. Obviously, the supply current arrives by different routes that are noted by ′ The total instantaneous current at the moment *t* consumed by the k-th voltage domain can be expressed as

$$i_{ck}(t) = \sum_{j=1}^{N} i_c^{(j)}(t) \,,$$


Primary power source
i_S
A
Conditioning circuit 1
Conditioning circuit 2
i_{c1}
i_{c2}
gnd
Delivery circuit 1
Delivery circuit 2
i_{c1}'
i_{c1}''
i_{c1}'''
i_{c2}'
i_{c2}''
Power sink: domain 1
Power sink: domain 2
Battery, transformer, energy harvesting element, USB, etc.
DC/DC converter, reference regulator, capacitors, etc.
PCB, wiring, capacitors, filters, etc.
Microcontroller, DSP, FPGA, ASIC, display, sensors, communication interface, wireless module, etc.

Generalized power delivery model

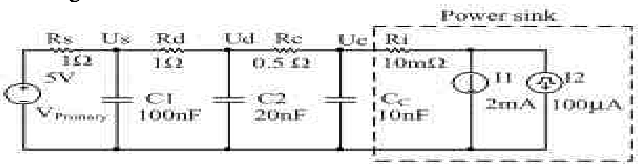
where *N* is the number of connected supply pins of the integrated circuit (IC). The current drawn from the primary source can be written as

$$i_s(t) = \sum_{k=1}^{K} (i_{ck}(t) + i_{Condk}(t)),$$

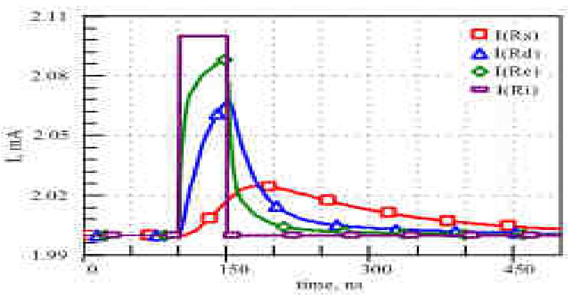
where *iCondk* is the current consumed for the operation of the conditioning circuit (for example voltage regulator, DC/DC converter, etc.). The total energy consumed from the primary source in the time interval from *t*1 to *t*2 can be expressed as

$$E = \int_{t_1}^{t_2} i_s(t) \cdot u_s(t) \cdot dt,$$

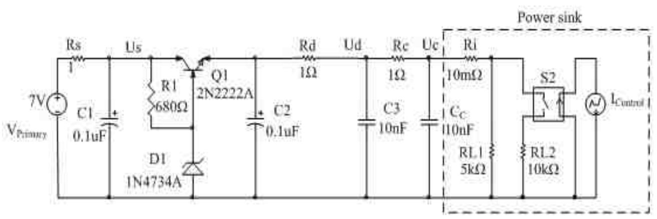
where *us(t)* is the instantaneous voltage between the poles of primary source. As power units nW are often used while µJ, nJ or pJ are used in case of energy characterization. Power units are usually estimated for long period averaged consumption, but energy units are used to characterizeshort events like energy consumed to execute an instruction of microcontroller. Time scale of power consumption measurement Dependant on the time interval $(t2 - t1)$ of energy or power estimation, averaged (coarse grained) or cycle (instruction)-accurate (fine grained) measurements are distinguished. If the time interval duration is not longer than the duration of icrocontroller's instruction execution time (related to the clock frequency) then measurements are treated cycle-accurate and can be used to estimate power consumed to execute individual instructions. Definition of cycle-accurate measurement is less easy for FPGAs that are devices executing program code not serially instruction- by-instruction but rather in parallel. The fluctuations of FPGA energy consumption will be influenced by the data processed in every clock cycle. The whole energy consumed by the ES is sourced from the primary source. Therefore, estimating energy from currents *ick* and *is* will produce the same results if averaged over a long period. Let us consider two typical power delivery schematics both complying with the model shown in Fig. 1. The first schematic represents power delivery without conditioning circuits, for example in case of battery supply. The second schematic represents circuit including standard voltage regulator composed of bipolar transistor and Zener diode. Voltage regulator outputs fixed supply voltage $V \approx 4.8V$ irrespective to input voltage from primary source in the specified voltage range. Both circuits include coupling capacitors that are recommended for the reduction of voltage fluctuations in supply lines. Supply lines themselves are represented by omic resistance *Rd* (delivery resistance). Resistor *Rs* represents wires connecting primary source to conditioning or delivery circuits. Resistors *Rd* and *Rs* on the other hand indicate potential locations of shunt resistor insertion for current measurement. The resistor *Rc* represents embedded microcontroller's or programmable logic device's supply pin connection resistance. The resistor *Ri* represents internal resistance included for the simulation of internal IC's current. In standard of the shelf boards like (for instance evaluation kits), coupling capacitors *CC* are often connected as close as possible to the chip supply pins. Thus, it is practically impossible to insert current measurement shunt or connect ammeter at the location of *Rc* . The same applies for IC that has multiple supply input pins, for example FPGA. To model instruction caused consumption current increase in Fig. 2 current source pulse I2 having duration of *ti* = 50*ns* is introduced at the moment *td* = 100*ns* . In Fig. 4 at the moment *td* to model the increase of consumed current the additional parallel resistor is connected for the duration *ti* using switch S2 controlled by the current pulse IControl. In Fig. 3 and 5 corresponding current waveforms are plotted. Current *I(Ri)* through the resistor *Ri* represents consumed current profile of the power sink (embedded microcontroller of programmable logic). Other currents *I(Rs)* , *I(Rd)* and *I(Rc)* through the respective resistances are shown on the same plots. The further is location of the resistor from the input pins of power sink, the stronger is low pass filtering influence of coupling capacitors. It can be concluded from the obtained simulation results that cycle accurate measurement is hardly possible while measuring current exiting primary supply source, i.e. *I(Rs)* . Current *I(Rs)* fluctuation is significantly slower and less in magnitude compared to *I(Rd)* especially in the circuit with voltage regulator because of large capacitances C1 and C2. For the cycle accurate measurements the whole energy related to the particular instruction should be measured during its execution to avoid overlapping with the energy consumption caused by adjacent instructions. If measuring *I(Rs)* one would be able to accumulate all energy only in much longer period as can be seen from both Fig. 3 and Fig. 5. Supply voltage fluctuations as seen from Fig. 5 are insignificant. Current consumption increase in voltage regulator *iCond* due to S2 switch modelled current increase is neglected in this simplified analysis. To implement cycle-accurate measurement it is vital to foresee current measurement as close as possible to the power sink of interest. In case of current measurement using shunt the resistor connection points in the supply route on the printed circuit board must be designed in advance.



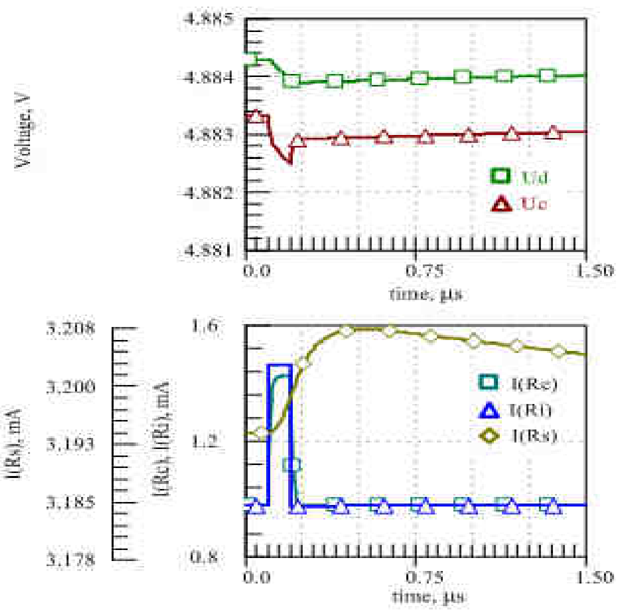
Example of power delivery schematics



Simulated current waveforms



Example of power delivery schematics with voltage reference



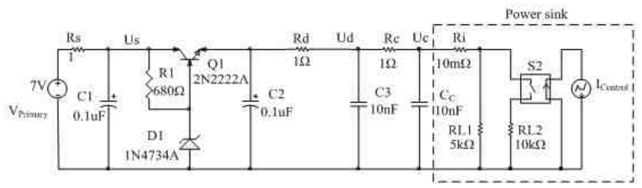
Simulated voltage and current waveforms

Power consumption measurement methods

Current shunt method

The most often used method is based on measuring voltage drop on the current shunt resistor inserted in the power supply line (high end) or ground (low end) line. Shunt resistor is usually selected in the range of tens to hundreds *mΩ* and the precision is not worse than 1 %. A similar approach is using a precision ammeter to measure supply current. A shunt resistor can be inserted in the appropriate supply line to measure either *iS* or *iCk* (see Fig. 1). Literature review revealed that measurement of the voltage drop on the shunt resistor is performed by one of the following:

1) using digital multimeter (DMM) in DC measurement mode; or 2) using standard or specially assembled data acquisition (DAQ) tools. Usually the second DMM or second channel of DAQ or oscilloscope is used to measure supply voltage level in order to calculate power or consumed energy.

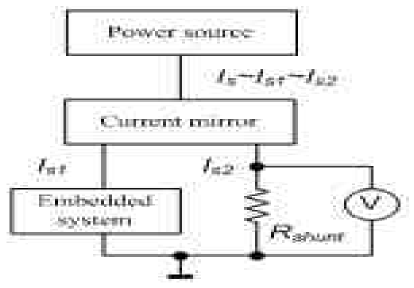


Example of power delivery schematics with voltage reference

Current shunt method is applied at both very low currents range, for example, to measure current of industry leading low power microcontroller MSP430 in the range of microamperes [5], and at the power hungry field of wireless communication modules like Bluetooth reaching hundreds of miliamperes .Back to 1994 Tiwari at. all used DMM to measure current consumption of Intel 486 microprocessors. To get stable readings they executed portion of program under investigation in an infinite loop since it was much less then averaging window of ammeter (100 ms). Measurement of power consumption was not cycle-accurate, but characterized the entire portion of the program. Cycle accurate power consumption measurement using 0.1Ω shunt resistor followed by the 275 MHz bandwidth differential amplifier is described in reference. Sampling frequency of 80 MHz and 8 bit ADC are used in the current measurement setup. Software source code is instrumented with special trigger points and is executed in real time on the target processor. Then logic analyzer (LA) is employed to track for specific trigger events in the system's address and data buses. LA simultaneously records trigger moments and current consumption samples from the ADC output. After the collection of data by LA current and voltage waveforms are back annotated to time stamps and energy consumed during the particular program segment or instruction is estimated off-line. The power of instructions is expressed in nWatts.

Current mirror method

A modification of supply current measurement method for power consumption characterization is presented in the series of works carried out by the group of Laopoulos. Authors utilize Wilson current mirror assembled from four bipolar transistors. It duplicates the current consumed by ES . A mirrored current *IS* 2 is then measured using shunt resistor. The advantage of this method is that shunt resistor is absent in the direct supply line, thus reducing influence to the measured system's supply voltage. Voltage drop on the shunt resistor then can be measured either with DMM, oscilloscope or DAQ means dependant on the time resolution requirements. To reduce the total current drawn from the primary power source the mirror current can be scaled a predefined number of times by appropriately selecting components in the schematics of the current mirror.



Current mirror method diagram

Current probe

Current probes exploiting magnetic coupling are widely used to measure higher currents due to security reasons. Though this is not the key issue in ES power consumption measurement but sometimes utilization of the current probes is reported for microprocessor systems power estimation. The obvious advantage of current probes is the absence of the need to cut supply wire in order to insert shunt resistor or ampermeter. However, probes can not be used to sense currents flowing in the PCB routes. In the case voltage regulator is assembled on the same PCB together with the rest of an ES, only the current entering regulator may be measured. It makes virtually impossible to get cycle-accurate results. Application of Agilent clamp-on current probes was in particular mentioned in the paper to explore Pentium 4 microprocessor based system power consumption. After sampling current probe's output with 10 kHz rate NI DAQ board the results were examined for the correlation to processor performance metrics. The EEMBC (Embedded Microprocessor Benchmark Consortium) consortium is also aware of the possibility to use nonintrusive current probes, such as Tektronix TCP312 for power estimation of ES. However, EEMBC offers current shunt method because of the much higher pricing of current probes .

Charge transfer method

Basically two types of methods can be distinguished:
1. Charging a known capacitor from the power supply, discharging by consumed current of the ES and measuring the remaining voltage, and
2. Charging a known capacitor by the consumption current and measuring the discharge time. In the paper a cycle-accurate energy consumption measurement system based on charge transfer using switched capacitors is presented. The idea is to power up the ES from charged capacitor for a short period (for example, during the execution time of the program's instruction of interest) and measure the voltage $V1$ at the beginning and the voltage $V2$ at the end of measurement period. Then the energy E in pJ consumed during the measurement period can be expressed

$$E = \frac{1}{2}C(V_1 - V_2)^2 ,$$

where C is the known capacitance in pF. The approach was validated over the DMM method and the measurement errors were found not to exceed 2–3 %. Authors then present the results of energy consumption measurement for the popular ARM7 microprocessor's instruction stages (address fetch, operation code encoding, operand access and execution). This time axis resolution was hardly ever mentioned in the other references. It must be also mentioned that the measurement setup requires high sampling frequency data acquisition means. Adoption of the current mirror to implement the second charge transfer method is discussed. Instead of driving mirrored current through the shunt resistor a capacitor C is charged for some time interval of interest. Then the capacitor C is switched off from the charging circuit and its discharge time is measured. The discharge time is proportional to the current consumed during the interval of observation. During the discharge phase of the capacitor C another capacitor is switched to be charged by the mirror current. Yet further modifications of the method eliminating expensive high speed data acquisition means are based on the counting of the number of times (pulse counting using electronic counter) the capacitor C is charged to the certain voltage level by the mirror current during the interval of observation. Charge transfer methods are suitable for cycleaccurate power consumption measurement though they are more complex to implement compared to the current measurement using shunt resistor.

Battery status monitoring method

Monitoring battery status is widely accepted in mobile and handheld equipment fields. Observation of the battery voltage drop after execution of the software portion of interest is also a measure of consumed energy. Usually the code is repeated many times to obtain the voltage drop that can be measured . Voltage drop per software function is calculated by simply dividing the total voltage drop from the number of times the code was executed. It must be mentioned that running the same function in a cycle may distort software related power consumption results because the code may be populated to the fast cache memory. It would not be the case if the analyzed code piece is invoked ones and the instructions are stored in slower external memory. The method is attractive due to its implementation simplicity. Acceptance of Smart Battery System (SBS) specification by many manufacturers standardizes monitored parameters and their transfer via SMSBus. Modern rechargeable batteries are equipped with embedded electronics compatible with the SBS. Battery voltage level very often can be obtained by invoking corresponding software function from the operating system of the mobile device. No hardware modification is needed. In the article the method for software power measurement based on correlating device's usage statistics (device states: sleep/operating, internal modules on/off) to the time averaged power consumption is explored using data obtained from SBS interface. The built model is then used to estimate instantaneous power consumption. Authors recommend providing unified Smart Battery measurements (currently vendors do not specify measurement averaging interval) and equip all system parts like CPU, PCI cards, disks with their own power measurement hardware.

Thermal monitoring method

Paper [7] presents an idea of how to estimate leakage power based on integrated circuit's (IC) thermal profile. The considered models relating leakage current and thermal profile are linear and of high accuracy in normal operating temperature range of many IC (55○C–85○C). Instead of physical temperature measurement chip thermal profile is estimated by specialized modeling software Hot- Spot3.0. It was shown that number of different temperature regions is not important in order to estimate leakage current with errors less than 3.5 %. By dividing the chip area in to 4 regions where temperature is estimated individually the errors can be reduced down to 0.5 %. Even though authors do not discuss possibility of chip temperature measurement (it can be noticed that sometimes manufacturers integrate temperature sensors inside a chip) the approach itself can be treated as an alternative method to others and needs further research. Surely, cycle-accurate consumption power measurement is not possible due to the very low dynamics of temperature as a physical quantity. Also, this method is not intended to estimate dynamic power which is caused by the switching activities inside a digital chip and most often constitutes the largest part of consumed power. However, increasing integration level inside modern microcontrollers and FPGA and entering nanometer scale in semiconductor industry leads to the considerable leakage currents. Another unexplored field of application is estimation of consumed power for the devices operating in so called sleep modes, where central processing unit is idle and switching due to data changes is minimized. Until now it was not investigated what accuracy of temperature estimation is needed in order to measure current changes bellow 1 μA which is usual in sleep modes. Industrial solutions and instruments Measurement equipment industries target ES power consumption measurement problems by introducing specialized tools to the market. To give an example the Agilent's DC Power Analyzer N6705A can be mentioned. In addition to manufacturer's white papers its application for FPGA based soft-microcontroller power estimation is described in paper. The Power Analyzer is capable of not only measuring current on the shunt resistor but also serves as a controlled multi channel voltage source which makes it very suitable for investigation of power consumption in systems having several supply voltage domains. Sampling frequency of this tool is 50 kHz which limits its application for cycle accurate power measurements of ES running at the clock rate of tens of megahertz. National Instruments offers its universal DAQ tools for the acquisition of current and voltage waveforms for consumption power estimation.

VII. FACTORS AFFECTING POWER CONSUMPTION OF A PROCESSOR

In order to devise new techniques for energy consumption reduction of the processor, knowledge about the factors affecting processor execution and influencing the performance is essential. There are a variety of factors on which power consumption of a processor depends on. It is hard to analyze the dependency of each factor on power consumption. In this paper, some of the significant factors are studied. The analysis needs to be started with CMOS circuit level. The two major types of power dissipations occurring in a CMOS circuit are:

- ***Static dissipation*:**

Due to leakage currents

- ***Dynamic dissipation*:**

Due to the charging and discharging of capacitance or due to the switching activities of the circuit. As switching activities increase with increased clock frequencies, processors with high operating clock will have more power dissipation. it is clear that the power onsumption of the processor depends linearly on the effective capacitance, supply voltage and clock frequency. Thus, by changing the voltage and frequency, the power dissipation of the processor can be controlled.

- **Reducing capacitance**

Reducing the parasitic capacitance of the circuit will result in power consumption reduction. Reduction of capacitance should be done with appropriate frequency scaling to get better results and performance increase. High frequency signals have to be routed through low capacitance and vice versa to conserve power. To reduce the capacitance for low power design, select a least gate size enough to meet the speed constraints of the signal switching.

- **Switching voltage**

Due to the quadratic relation of voltage term, reducing the supply voltage can result in substantial savings. The well known technique is the Dynamic Voltage Scaling (DVS) used to save power [6]. But, when the supply voltage is reduced to conserve power, there are some major concerns to be addressed. The performance will vanish due to the slow CMOS transistor. This may lead to system failure in meeting the timing constraints which may result in catastrophic effects for hard real time systems [7]. Therefore, enough precaution must be taken to ensure that power reduction is achieved without violating the temporal and functional constraints of the system. Also, at low voltages, noise immunity becomes difficult.

- **Clock frequency**

Reducing the clock frequency is beneficial for power consumption reduction and has the same effect as that of reducing the capacitance of the circuit. Eliminating the unwanted logic switching will help to reduce the clock frequency. The other factors that can help to reduce the switching frequency are: changing the number representation and coding forms, finding an alternate logic design for the same circuit, etc. Reducing the switching frequency will improve the system reliability.

- **Frequency switching**,

Preemption and Context switches the main factors on which the processor power dissipation directly depends on are the load capacitance, supply voltage and clock frequency. For low power consumption, the processor voltage and frequency has to be reduced without violating any other specifications. The key technique for power consumption reduction of an embedded processor is Dynamic Voltage Frequency Scaling (DVFS), in which the processor is operated at different operating points which is a duple of supply voltage, clock frequency. But, frequent switching between the operating points is also not desirable, as power wastage is associated with each switching. Another factor which if not controlled can lead to an increase in power consumption of the processor is the preemptions and context switches between tasks or processes. Preemptions and associated context switches are essential for the implementation of an embedded application. If preemptions are not allowed, when a high priority task is ready to execute, it may not get the control of the processor due to execution of a low priority task. When preemption occurs, the context of the task namely, the program counter, temporary register values, etc. are to be saved into the stack. This is context switching and if these are not controlled can even fail the schedulability of the system and can lead to considerable amount of energy wastage.

- **Cache misses**

Cache is the smallest and the fastest memory unit associated with every embedded device. The essential data which is needed for the future computations are stored in the cache and is used by the processor. But, due to the limitation of the memory capacity, all the data cannot be stored as required by the processor, and can lead to cache misses. Cache misses have a great impact on the processor energy consumption due to the additional clock cycles wasted to find the data and the delay incurred. This leads to an increased instruction time and will add to energy penalty.

- **Resource constraints**

Any system design will focus on how effectively the resources can be utilized to get the maximum or optimal performance. Always there exists a tradeoff between the performance achievable and the available resources. The major impact of limited resources is the stalls introduced in the pipeline which degrades the performance of the system, by the inclusion of additional instruction cycles needed to complete the set of instruction execution. Therefore, this factor also has an adverse effect on the energy consumption

- **Branch Prediction Overhead**

In order to increase the performance of processors with any architecture support, or especially in multi-core systems, branch prediction technique is widely used for reducing the overhead associated with conditional instruction execution. Every branch instruction has to have a penalty of few clock cycles to find the correct address to which the Program Counter (PC) should point next. This is one of the major pipeline hazards which introduces stalls in the pipeline and increase the energy wastage of the system.

- **Insertion of *WAIT* States**

Wait states are inserted in a code when there is an off-chip memory access. When some data has to be accessed from an external memory unit, the delay associated will be more and to incorporate that into the program code, WAIT states has to be inserted, due to the lower operating frequency of the processor execution.

- **Register File Size**

The register file size i.e. the number of registers required by a program code while executing an application task, has an adverse effect on power consumption. As the register file size increases, the amount of memory usage increases, increasing the chip area. Increase in chip area will increase the power consumption of the processor. The use of an optimum number of registers will result in shorter instruction word to reduce the power consumption.

- **Power-aware Instruction Selection**

Another factor which can be considered for power consumption reduction is the selection of appropriate instruction sequence. By identifying a power-aware instruction sequence and avoiding costly external memory accesses, some wasteful power consumption can be reduced.

- **Program execution time**

The execution time or computation time of a code has a direct influence on the energy consumption of the processor. As the code size increases, the number of processor cycle also increases; increasing the duration of active state of the processor. Being in the active state, the processor consumes more energy when compared to an idle state. Therefore, code optimization becomes essential for any embedded application development.

- **Instruction Scheduling**

By re-ordering the instruction execution, without changing the functionality, some amount of power saving can be claimed. By doing so, the circuit can be made to retain in similar states with slight changes only. With reduced state changes, the circuit saves some power dissipation. Along with instruction scheduling, other techniques like: register pipeline, memory layout, jump optimization etc can also be used for better power saving .

- **Impact of Real Time Operating System (RTOS)**

Real time operating systems are used in most of the real time embedded applications to manage the software and hardware resources. Due to this nature, RTOS dissipates a significant amount of power in the system. It was found that on an average; around 32% of the total energy was drawn by the OS alone for a particular workload . But, the impact of RTOS to the application and to the processor/controller is mostly neglected. The user gets estimation about the computation time of different parts of the RTOS when a particular hardware configuration is used. But, there is no knowledge about the power impact. Actually, there is an effect of use of RTOS on the performance and power consumption of the system. The major power consumtion happens when the RTOS is running the application software. From the literature, it can be observed that the percentage of energy consumed by the RTOS and the Board Support Programs (BSP) can vary on average from 1% to 99% based on the level of software dependence on the RTOS.

- **Switching of processor states**

Over the time of execution, there is a continuous transition of the processor state, which is a factor to be considered for power consumption. For energy saving, normally processors are assigned with different states like stand-by, power down, idle, sleep, nap etc. When the processor is not executing any job, it is switched to sleep, stand-by or any other idle state to save energy. But, there is an average of some few milliseconds needed to switch the processor from an idle state to an active state. So, by incorporating some predictive power management techniques in the processor, the future re-activation time can be predicted well in advance to avoid the delay and associated power wastage.

- **Mass Data Storage and Handling**

While implementing large and complex applications, huge amount of data is needed to be handled and also stored. This is highly energy inefficient. Generally in data grid systems, the data storage and data computing units are placed far away, and therefore, there is a necessity for huge data transfer. As an alternative, data are replicated at many different locations for reducing the accessing time; avoid loss of data and to reduce the huge amount of data transfer. So, there should be good techniques to support data placement and replacement to have better energy efficiency.

- **Addressing modes**

Any processor architecture will support different addressing modes to facilitate ease of data/operand fetch. The different addressing modes allow the processor to fetch the data in various manners to ease the data procurement and the coding. But, there is an influence of different addressing modes on the power consumption of the processor. One reason is the variations in the length of code aiding the power consumption when the processor expends more time to find the data for executing a particular instruction.

- **External memory access**

Embedded systems, being developed for meeting specific functionality, the associated resources of the system are also limited. Most of the embedded systems have limited on-chip memory including the caches. But, when running complex and large applications, the memory requirement will be more, leading to external memory accesses, which is a major source of energy consumption in embedded systems. The reason is external memory accesses incur additional energy dissipation in terms of increased memory access time and delay. So, during the development stage of the application, enough measures have to be taken to fix the memory capacity of the system. The execution time of a set of code depends on the values of input data. Similarly, there is a firm dependence of energy consumption of different instructions on the data on which it is operating. Because, based on the input data, the conditional instructions which will be executing and therefore the total number of instructions executed will vary; varying the energy consumption. The other energy susceptible parameters are: the value of operands, operand and fetch address of the instruction, register number and values.

- **Transistor size**

Transistors are the basic building block in circuit level and the main factor that influences the power dissipation, area and performance of the circuit is the size of the transistor. Reducing the transistor size is beneficial because it results in a linear reduction in capacitance, thereby reducing the dynamic power . Also, on analyzing the signal probabilities, transistor restructuring in a combinational circuit can help to achieve superior power efficiency. Small techniques like equivalent pin reordering can be exploited because of the fact that logically identical pins may not have similar delay or power consumption. Another important component which is very frequently used in synchronous systems is flip flops and latches. A large amount of power is dissipated to operate latches and flip flops, so reducing the clock power dissipation of these elements can conserve some power.

- **Clock signals**

At the device level, the major power dissipation which occurs in CMOS circuit is the *capacitive power dissipation.* Due to the charging and discharging of node capacitance, *internal switching power* is the dynamic power dissipated inside the logic cell which is a sum of charging and discharging of internal nodes and the short circuit power. Third is *static power,* which is based on the state of the logic circuit. These power dissipations depend on various factors like: operating voltage, temperature, signal slope, output load capacitance, fabrication process etc. Also, some other factors that influence the power dissipation is the logical encoding, Boolean function description and information representation. Clock signal contributes to around 40% of total system power dissipation. Different techniques like clock gating, reduced swing clock etc. can be adopted to reduce the power dissipation associated with the clock signals.

- **Correlation between samples**

Register Transistor Level is also known as architecture level or block level. In this level, the high level functions are performed by the basic units such as: registers, buses, multiplexer, adders, multipliers, memories, state machines etc. In this level, power dissipation can be expressed as a function of the number of bits of the components and the operating frequency. It does not consider the data dependency of power dissipation. The correlations between successive samples are beneficial because there may be more number of bits which are in common. In some cases negative correlation also exist. Anyway, the correlation has an impact on power dissipation because of the switching activities of the data path. The switching activities has to be reduced as minimum as possible to conserve power. Buses are the mode of data exchange between different parts of the circuit; it is a major source of power dissipation in digital circuits. Therefore, applying reduced voltage swing can help to reduce the power dissipation.

- **Memory**

Memory is a vital part of any circuit and the technological growth has allowed the integration of more on-chip memory and data caches. For low power dissipation, bulk Random Access Memory (RAM) units have to be operated at low voltages. But this may affect the speed of operation of the system and to balance, the threshold voltage can be reduced. Better means like using multiple threshold devices, dynamic adjusting of threshold voltage by back bias voltage control, memory bank partitioning etc. can be adopted . Memory unit is considered as one of the greatest consumers of power. For example, in the case of STRONG ARM controller, the cache consumed around 43% of the total power.

- **Exploiting parallelism**

Different power management techniques are also in common for conserving power like using different power down modes, deactivating the part of the circuit which is not functional etc. Another attractive technique is to exploit parallelism which helps to reduce the clock frequency and supply voltage of the system. Pipeline techniques are now very versatile and a uni-processor system with pipeline is *n* times better to improve the power efficiency. Loop unrolling techniques can aid in lowering the operating voltage and clock frequency. The factors discussed are only some of the factors which has an impact on power consumption. It is also desirable to consider power consumption issues at different levels of abstraction. The following section gives a discussion on the power reduction techniques possible at different levels of abstraction.

VIII. DIFFERENT LEVELS OF ABSTRACTION AND POWER REDUCTION TECHNIQUES

The power minimization can be attempted at different levels of abstraction viz, at circuit level, device level, logic level, architectural level and at algorithmic level. A brief description about each level of abstraction is included in the following section.

- **Algorithmic level**

In this higher level of abstraction, energy minimization is attempted by reducing the number of operations by analyzing the cost of each operation. It mainly includes: by comparing the arithmetic operations and logical operations, cost of memory accesses and incorporating ways to maximize the spatial and temporal locality of references. Also, there can be energy saving by considering the number representations and selection of different data representations like two's complement, sign magnitude etc. Another similar factor that can be beneficial is the bit length. Power reduction can be achieved at the cost of accuracy loss in the results.

- **Architectural level**

In the architectural level, different techniques for instruction set design and performance increase by software approaches can be dealt with. The major in this group is exploiting the parallelism at instruction level, thread level and data level. Also, another widely implemented technique is pipelining; which can be useful for power minimization at the architectural level. Here, processor voltage is lowered to conserve power and then parallelism is exploited for enhancing the performance of the system. When the processor supply voltage is reduced, the speed of the system will come down. To supplement this, software techniques like, pipelining can be used to maintain the throughput of the system. There are many other software techniques that can be possibly incorporated into the architecture: speculation and prediction, loop unrolling, lop termination, procedural inlinig, product specific selection, global variable localization and assembly language in lining etc. to save energy and to enhance parallelism.

- **Logic and Circuit level**

At the logic and circuit level, methods namely reducing the voltage swing, reducing the effective load capacitance can be done to reduce the power dissipation. Also, due to the variations in the signal arrival time to the input of a gate, glitches may occur which will add to power consumption. Therefore, logic restructuring and path delay balancing techniques can be used to reduce the glitches to save power. Mostly, static logic circuits are the sufferers of glitches when compared to dynamic logics. Also, high probability switching nodes are operated at reduced speed or hided to reduce the switched capacitance to conserve power. Use of hybrid library consisting of static CMOS gates and pass transistors for synthesis are found beneficial for power reduction. Equivalent pin reordering can be used for power reduction along with transistor reordering technique. Clock power minimization can be achieved with techniques like: clock gating, low swing clocking and clock distribution minimization to save power dissipation.

- **Device level**

Threshold voltage reduction or dual threshold techniques can be used to reduce the leakage power dissipation. Use of high threshold devices for non-critical delay paths and low threshold devices for critical path can result in standby power consumption reduction. The factors which are mentioned above include only a subpart of a variety of parameters on which the power consumption of the processor depends. There can be more factors which may influence the power dissipation of an embedded processor. But, the above discussed parameters are a few, which are essential to be known and considered during the design phase or later. Most of the above referred elements are software techniques which can be easily adopted while developing the program code. The power conservation can be attempted well with software techniques rather than hardware approaches. Therefore, which technique is to be used for a particular processor architecture has to be identified aptly by the system designer to alleviate the power dissipation of the processor.

IX. CONCLUSION

In this paper we produced the reasons and factors that affect on power consumption in embedded system Processors starting from the definition of Embedded system.

REFERENCES

[1] V. Tiwari, S. Malik, and A. Wolfe, “Power analysis of embedded software: A first step towards software power minimization,” *Proc. Int. Conf. Computer-Aided Design*, Nov. 1994.

[2] C.-T. Hsieh, M. Pedram, G. Mehta, and F.Rastgar, “Profile-driven program synthesis for evaluation of system power dissipation,” *Proc. Design Automation Conf.*,June 1997.

[3] C-T. Hsieh and M. Pedram, “Micro-processor power estimation using profile-driven program synthesis," *IEEE Trans. on Computer Aided Design*, Nov. 1998.

[4] L. Benini and G. De Micheli, “System-level power optimization: Techniques and tools,*” Proc. Int. Symp. Low Power Electronics and Design*, Aug. 1999.

[5] T. Simunic, G. De Micheli, and L. Benini, “Energyefficient design of battery-powered embedded systems,” *Proc. Int. Symp. Low Power Electronics and Design*, Aug. 1999.

[6] J. L. da Silva, F. Catthoor, D. Verkest, and H. De Man, “Power exploration for dynamic data types through virtual memory management refinement,” *Proc. Int. Symp. Low Power Electronics and Design*, Aug. 1999.

[7] V. Tiwari, S. Malik and A. Wolfe, “Instruction level power analysis and optimization of software,” *Journal of VLSI Signal Processing*, 1996.

[8] C.-H. Hwang and A. Wu, “A predictive system shutdown method for energy saving of event-driven computation*,” Proc. Int.. Conf. on Computer Aided Design*, November 1997.

[9] E. Chung, L. Benini and G. De Micheli, “Dynamic power management for non stationary service requests", *Proc. Design and Test in Europe Conference*, March 1999.

[10] Q. Qiu, Q. Wu and M. Pedram, “Stochastic modeling of a power-managed system: Construction and optimization,” *Proc. Symp. on Low Power Electronics and Design*,, August 1999.

[11] T. Simunic, L. Benini, and G. De Micheli, “Dynamic Power Management of Laptop Hard Disk", *Proc. Design Automation and Test in Europe*, 2000.

[12] Q. Qiu, Qing Wu, and M. Pedram, “Dynamic Power Management of Complex Systems Using Generalized Stochastic Petri Nets”, *Proc. the Design Automation Conference*, Jun. 2000.

[13] H. Mehta, R. M. Owens, and M. J. Irwin, “Some issues in gray code addressing,” *Proc. Great Lakes Symposium on VLSI*, Ames, IA, USA, Mar. 1996.

[14] W-C. Chung and M. Pedram, “Power-optimal encoding for DRAM address bus,” *Proc. Symposium on Low Power Electronics and Design*, July 2000.

[15] K. Kim and P. A. Beerel, “A low-power matrix transposer using MSB-controlled inversion coding,” *The First IEEE Asia Pacific Conference on ASIC*, 1999.

[16] L. Benini, G. DeMicheli, E. Macii, M. Poncino, and S. Quer, “System-level power optimization of special purpose applications: The beach solution,” *Proc. Symposium on Low Power Electronics and Design*, August 1997.

[17] M. R. Stan and W. P. Burleson, “Limited-weight codes for low-power I/O,” *IEEE Transactions on VLSI*, Mar. 1995.