Microarchitecture-Level SoC Design

27

Young-Hwan Park, Amin Khajeh, Jun Yong Shin, Fadi Kurdahi, Ahmed Eltawil, and Nikil Dutt

Abstract

In this chapter we consider the issues related to integrating microarchitectural IP blocks into complex SoCs while satisfying performance, power, thermal, and reliability constraints. We first review different abstraction levels for SoC design that promote IP reuse, and which enable fast simulation for early functional validation of the SoC platform. Since SoCs must satisfy a multitude of interrelated constraints, we then present high-level power, thermal, and reliability models for predicting these constraints. These constraints are not unrelated and their interactions must be considered, modeled and evaluated. Once constraints are modeled, we must explore the design space trading off performance, power and reliability. Several case studies are presented illustrating how the design space can be explored across layers, and what modifications could be applied at design time and/or runtime to deal with reliability issues that may arise.

Acronyms

AHB	Advanced High-performance Bus
APB	Advanced Peripheral Bus
ASIC	Application-Specific Integrated Circuit
BER	Bit Error Rate
BLB	Bit Lock Block

Y.-H. Park
Digital Media and Communications R&D Center, Samsung Electronics, Seoul, Korea
e-mail: younghwp@uci.edu

A. Khajeh
Broadcom Corp., San Jose, CA, USA
e-mail: amin.khajeh@broadcom.com

J. Yong Shin • F. Kurdahi (✉) • A. Eltawil • N. Dutt
Center for Embedded and Cyber-Physical Systems, University of California Irvine, Irvine, CA, USA
e-mail: junys@uci.edu; kurdahi@uci.edu; aeltawil@uci.edu; dutt@uci.edu

S. Ha, J. Teich (eds.), *Handbook of Hardware/Software Codesign*,
DOI 10.1007/978-94-017-7267-9_28

CA	Cycle Accurate
CDMA	Code Division Multiple Access
CMOS	Complementary Metal-Oxide-Semiconductor
CMP	Chip Multi-Processor
CPU	Central Processing Unit
DFS	Dynamic Frequency Scaling
DMA	Direct Memory Access
DTA	Dynamic Timing Analysis
DTM	Dynamic Thermal Management
DVFS	Dynamic Voltage and Frequency Scaling
DVS	Dynamic Voltage Scaling
ESL	Electronic System Level
GPIO	General-Purpose Input/Output-pin
IDC	Inquisitive Defect Cache
IP	Intellectual Property
IPB	Intellectual Property Block
ISS	Instruction-Set Simulator
ITRS	International Technology Roadmap for Semiconductors
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor
MPSoC	Multi-Processor System-on-Chip
MTF	Mean Time to Failure
NMOS	Negative-type Metal-Oxide-Semiconductor
PDF	Probability Density Function
PI	Principal Investigator
PMOS	Positive-type Metal-Oxide-Semiconductor
PSNR	Peak SNR
RAM	Random-Access Memory
RDF	Random Dopant Fluctuations
ROM	Read-Only Memory
RTL	Register Transfer Level
SNR	Signal-to-Noise Ratio
SoC	System-on-Chip
SRAM	Static Random-Access Memory
SSTA	Statistical Static Timing Analysis
T-BCA	Transaction-based Bus Cycle Accurate
TLM	Transaction-Level Model
VFI	Voltage/Frequency Island
VOS	Voltage Over Scaling
WCDMA	Wideband CDMA

Contents

27.1 Introduction

A typical System-on-Chip (SoC) is shown in Fig. 27.1. There are four major complicated heterogeneous components in SoCs, such as *processors* (ARM7 shown in the figure), *custom hardware Intellectual Property Blocks (IPBs)* (memory controller, DMA controller, interrupt/GPIO controller and so forth), *on-chip memories* (RAM and ROM) and *on-chip communication architectures* (AHB and APB bus [8]).

These components have their own role such as processors run embedded software and usually control overall operation, custom hardware IPBs are dedicated to execute particular tasks, memories are storage place for data and instructions to be used, and all of which are connected through a on-chip communication architecture consisting of multiple shared interconnected buses using a specific arbitration scheme for fair sharing of the limited bus bandwidth.

27.1.1 A Typical System-on-Chip Design Flow

In recent years, research in this field has focused on the problem of defining a framework for SoC design that promotes Intellectual Property (IP) reuse, with particular attention paid toward achieving performance goals. Such a framework needs to have clearly defined abstraction levels for capturing the SoC design. The basic idea is to model the system first at a high level of abstraction, and then gradually refine the model to create models with higher levels of detail, until we arrive at the gate-level model (netlist). The SystemC [4] or SpecC [25] methodology

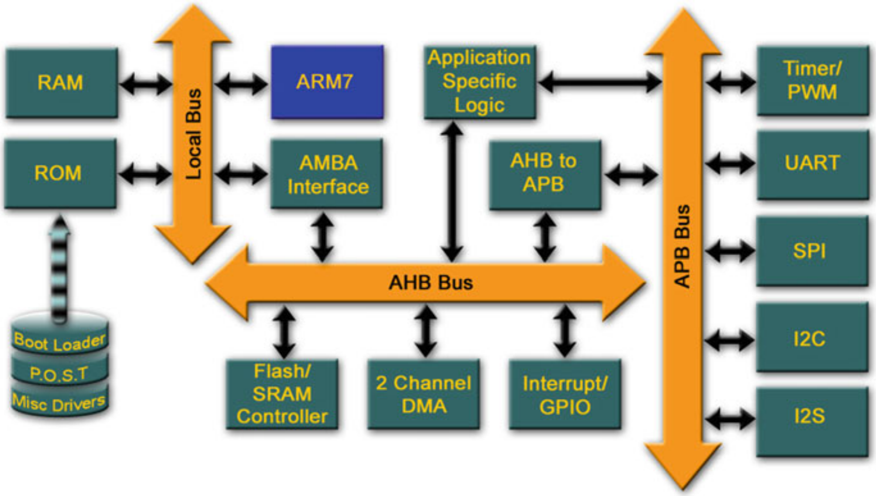


Fig. 27.1 An example of an SoC

focuses on defining a framework in which the system is initially captured at the specification level, and then gradually refined to generate models at lower levels of abstraction. This framework allows reuse of protocol libraries and IPBs at various levels.

Figure 27.2 outlines the typical flow of SoC design in terms of the levels of abstraction at which the designer can simulate the performance of an SoC, and perform communication architecture exploration.

The modeling abstraction levels in Fig. 27.2 are typically used for communication space exploration, with the application/algorithm usually captured with high-level languages such as C/C++. In Cycle Accurate (CA) models, the bus architecture and system components (both masters and slaves) are captured at a cycle and signal accurate level. While these models are extremely accurate, they are too time-consuming to model and only provide a moderate increase in speed over Register Transfer Level (RTL) models. Recent research efforts have focused on using concepts found in the domain of Transaction-Level Models (TLMs) to speed up simulation. Transaction-level models are very high-level bit-accurate models of a system, with specifics of the bus protocol replaced by a generic bus (or channel), and where communication takes place when components call read() and write() methods provided by the channel interface. Since detailed timing and signal accuracy are omitted, these models can be simulated quickly but are only useful for early embedded software development and high-level functional validation of the system. Transaction-based Bus Cycle Accurate (T-BCA) models overcome the slow simulation speed of CA models and the low accuracy of TLMs. T-BCA models capture timing and protocol details, but model components at a less detailed behavioral level, which allows rapid system prototyping and considerable

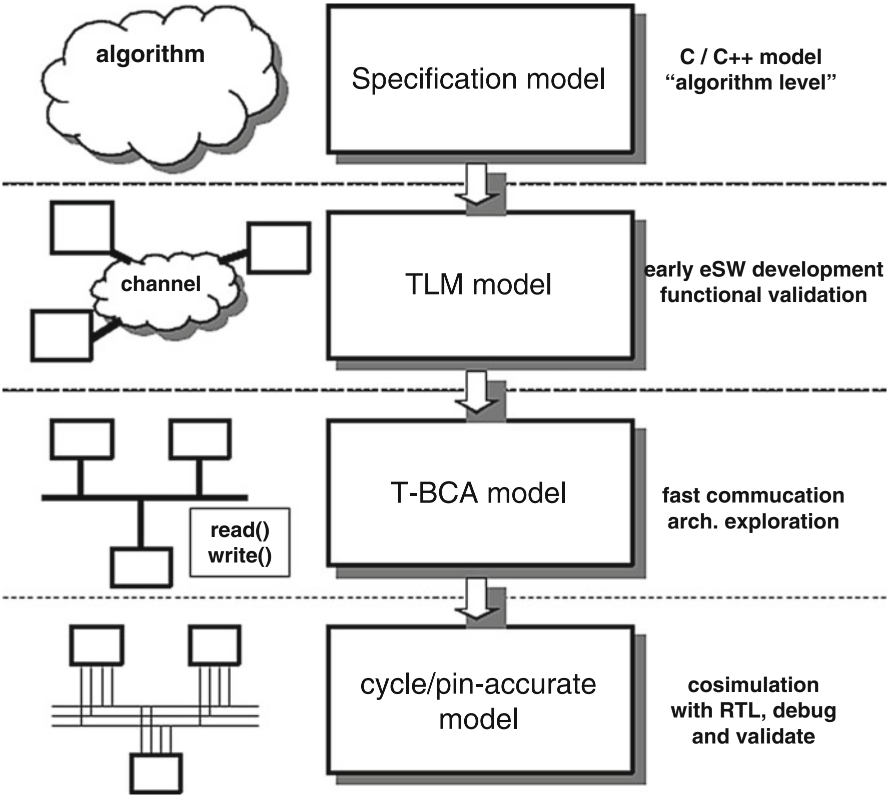


Fig. 27.2 A typical flow for an SoC design [105]

simulation speed over RTL. The component interface and the bus, however, are still modeled at a cycle-accurate level, which enables accurate communication space exploration.

27.2 Power Modeling

Reducing power dissipation is a critical design goal for electrical devices, from handheld systems with limited battery capacity, to large computer workstations that dissipate vast amounts of power and require costly cooling mechanisms. Ever-increasing performance needs, which require large parallel processing at fast clock frequencies accessing huge amounts of data from on and off-chip memories through a very complicated bus interconnection architecture, make this power issue more critical. In reality, their power densities and associated heat generation are exponentially increasing, as shown in Fig. 27.3 [5].This rapid increase of power

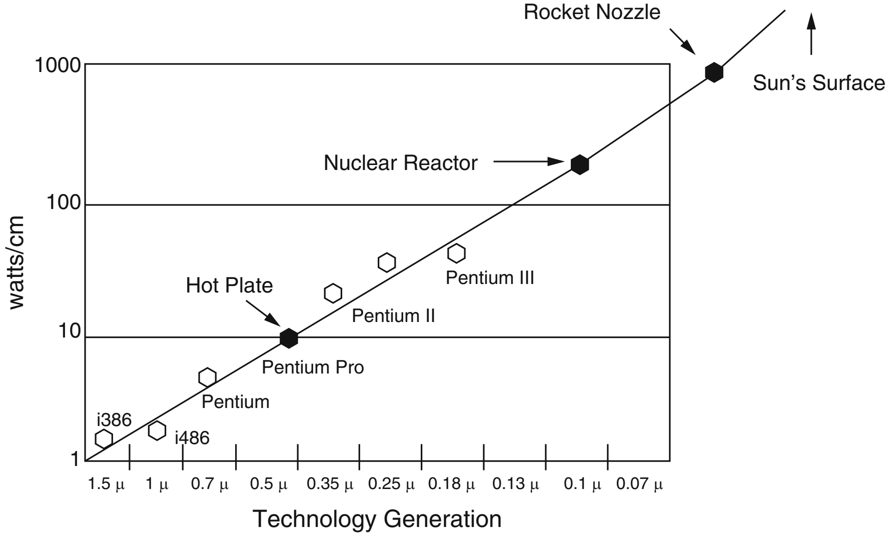


Fig. 27.3 The increase of power densities [5]

dissipation is of great concern, not only because of the aforementioned usage time of handheld devices and cooling costs but also because it can cause many unfortunate side effects, such as damaging chip reliability and reducing expected life cycles.

To address the ever-widening designer productivity gap, TLMs [20,82] and high-level simulation platforms are increasingly being used for SoC architecture analysis and optimization. The increasing importance of power as a design objective in today's complex systems is making it imperative to address power consumption early in the design flow, at the system level, where the benefits of performing power optimizing design changes are the greatest. Since design changes are easier and have the greatest impact on application power dissipation at the system level [58, 109], designers today must evaluate various power optimizations as early as possible in an Electronic System Level (ESL) design flow. In order to explore these optimizations, accurate power estimation models are necessary. These models are especially important for Chip Multi-Processor (CMP) systems with tens to hundreds of processors integrated on a single chip. Even a slight inaccuracy in power estimation for a single processor can result in a large absolute error for the chip. Several system-level power estimation approaches have been proposed in recent years, focusing on the various components of CMP designs, such as processors [58], memories [42], interconnection fabrics [84], and custom ASIC blocks [80]. Because of the heterogeneity of these components, power estimation models are usually customized for each component to achieve desired estimation accuracy. In addition, each type of component requires several power estimation models that can be incorporated at the most coarse grain, high levels of abstraction, as well as at the most detailed, low-level simulation abstractions.

27.2.1 Sources of Power Consumption and Defining Energy

In digital Complementary Metal-Oxide-Semiconductor (CMOS) circuits, there are three key sources of power consumption, shown below [87]

$$P_{total} = P_{dynamic} + P_{short-circuit} + P_{leakage} \tag{27.1}$$

Decomposed equations for each source, respectively, can be described by the following equations. Firstly, dynamic power consumption can be derived as below:

$$P_{dynamic} = \alpha_{0->1} C_L {V_{dd}}^2 f_{clk} \tag{27.2}$$

where $\alpha_{0->1}$ is the probability that a power consuming switching occurs, C_L is the load capacitance, V_{dd} is the supply voltage, and f_{clk} is the clock frequency. Note that Eq. 27.2 shows an important characteristic of dynamic power, which is that the power is quadratically proportional to the supply voltage, and can be efficiently reduced as the supply voltage level is reduced.

Secondly, the short circuit power consumption is formulated as below:

$$P_{short-circuit} = I_{short-circuit} V_{dd} \tag{27.3}$$

where V_{dd} is the supply voltage, and $I_{short-circuit}$ is the short circuit current which arises when both the NMOS and PMOS transistors are concurrently turned on, making a direct path from the supply power to ground. However, since this short circuit power consumption is responsible for only 10–15% of total power consumption and researchers have not found a good way to reduce this power without sacrificing performance [87], we will not focus on this component of power consumption in detail.

Finally, leakage power consumption can be calculated by:

$$P_{leakage} = I_{leakage} V_{dd} \tag{27.4}$$

where $I_{leakage}$ is the leakage current and V_{dd} is the supply voltage. Besides the dynamic and short-circuit power, transistors also consume leakage power (also referred to as static power), which is quickly becoming the large portion of total power consumption, based on the recent International Technology Roadmap for Semiconductors (ITRS) 2008 update [5], as shown in Fig. 27.4. Unlike dynamic power, the leakage power consumption continues during logic's idle status, and most of the techniques for dynamic power saving are not helpful for leakage power saving [5]. There are two major sources of these leakage currents, which are subthreshold leakage and gate-oxide leakage.

The first major component of the leakage current is gate-oxide leakage current, which flows from the gate of a transistor into its substrate. The thickness of the oxide

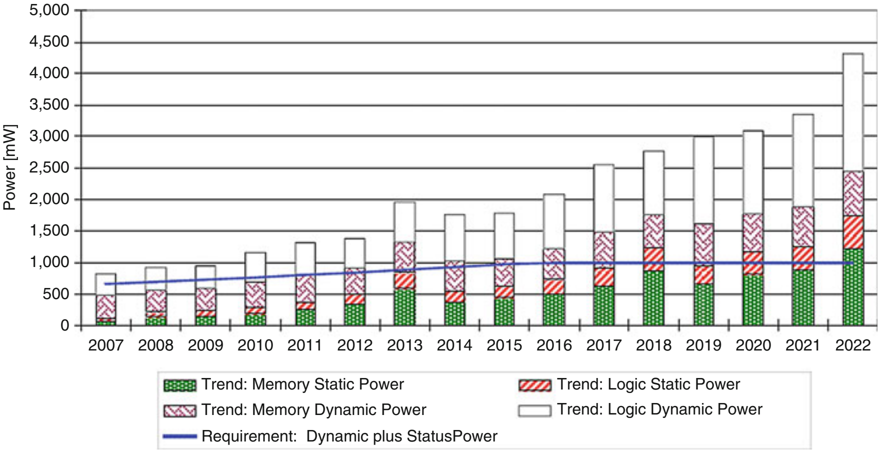


Fig. 27.4 Power consumption trends of portable chips [5]

material that insulates the gate decides this leakage current. The equation [87] for this type of leakage is given by:

$$I_{gate-oxide} = K_1 W \left(\frac{V_{dd}}{T_{ox}}\right)^2 e^{-\alpha \frac{T_{ox}}{V_{dd}}} \tag{27.5}$$

where W is the gate width, T_{ox} is the thickness of the oxide, V_{dd} is the supply voltage, and K_1 and α are constants. Based on Eq. 27.5, the gate-oxide leakage, $I_{gate-oxide}$ will be increased exponentially by decreasing the thickness (T_{ox}) of the oxide material of the gate. In conjunction with other design parameters, such as transistor sizes and supply voltage, the thickness of the transistor (including T_{ox}) for the upcoming chip design will also be decreased, and this will cause the exponential increase of the gate-oxide leakage current. Insulating the gate with high-k dielectric material might be the best possible solution for the problem of increasing gate-oxide leakage for the next few years [5].

The second major component of the leakage current is subthreshold leakage current, which flows between the drain and source terminal of a transistor. When the gate-source voltage, V_{gs}, exceeds the weak inversion point, it is still lower than the threshold voltage, V_t, and the MOSFET works like a bipolar transistor. The subthreshold current in this region changes exponentially, depending on the gate-source voltage, V_{gs}. The current in this subthreshold region is formulated by [5]:

$$I_{subthreshold} = K_2 W e^{\frac{-V_t}{nT}} \left(1 - e^{\frac{-V_{dd}}{T}}\right) \tag{27.6}$$

where W is the gate width, V_t is the threshold voltage, V_{dd} is the supply voltage, T is the temperature, and K_2 and n are constants. According to the Eq. 27.6, when the threshold voltage, V_t, is reduced, the subthreshold leakage current is increased exponentially. In response to increasing requirements for reducing technology scale parameters for upcoming chip designs, the threshold voltage should be reduced in conjunction with the supply voltage, and this causes a worse problem of subthreshold leakage. The increased subthreshold leakage current can cause another serious problem, called thermal runaway. A vicious cycle can result, in which the increased leakage currents cause increased temperature, then the increase temperature again causes more leakage currents, based on Eq. 27.6.

On the other hand, energy can be defined by the total quantity of the work a system completes over a period of time and formulated as the following equation :

$$E(\mathscr{T}) = \int_0^{\mathscr{T}} P(t)dt \tag{27.7}$$

where E is energy, *P(t)* is the instantaneous power at time t, and T is a time interval. The unit for energy is joules (J), and the unit for power is watts (W). In a computing system, power is the rate at which the system consumes the supplied electricity while performing computing activities, and energy is the total amount of consumed electricity over time for the task [87].

Note that some techniques to decrease power do not always decrease energy consumption. For instance, the power used by a computing system can be reduced to half by halving the supplied clock frequency; however, the total energy consumed will be approximately the same, because computing time will be twice as long to run the same task.

Different approaches will be necessary for reducing power or for reducing energy, depending on the context. For a system (e.g., a workstation) in which temperature is an important concern (because high temperature can cause various problems such as decreasing the overall speed of chips, increasing cooling costs, damaging chip reliability, and causing the thermal runaway problem), we must reduce instant power, despite the influence on overall energy, to keep the temperature of the system within tolerable limits. In handheld systems, however, reducing energy is usually the more important issue, because it is directly related to the battery lifetime.

27.2.2 Overview of Power Saving Techniques

There are many proposed techniques to reduce the power and energy of digital systems. The discussion in this section provides an overview for the most widely used power saving techniques.

Approaches for minimizing power consumption in CMOS digital systems involve various design abstraction levels, from the software algorithm and architectures to circuits. Some important power saving techniques are summarized in Table 27.1. They are classified by enabling time and sources of targeted power consumption (leakage/dynamic). Some techniques can be employed at the time of

Table 27.1 Summary of power saving techniques

Power	Design time	Idle time	Run time
Dynamic	Lower Vdd Multi-Vdd Transistor sizing Logic optimizations	Clock gating Operand isolation	Dynamic voltage Scaling (DVS) Dynamic frequency Scaling (DFS)
Leakage	Multi-Vt	Sleep transistors Multi-Vdd Variable Vt	Variable Vt

design, such as modification of transistor size and logic optimization, while other techniques, including varying supply voltage, clock frequency, and threshold voltage, can be either implemented statically at the design time or applied dynamically during the run time [84].

There are several techniques to decrease dynamic power consumption in particular. These techniques have different trade-offs, and some of them do not necessarily reduce the total energy consumption.

Clock gating (CG) is a widely used power optimization technique that saves dynamic power by stopping clock supply to unused portions in synchronous logic designs. "*Multistage clock gating*" refers to the scenario in which a clock gating cell controls either another one, or an entire row of clock gating cells. The synthesis tool identifies common *enables* and groups them with another clock gating cell. This technique is used for further optimization of existing gating cells by merging more register banks, so that the clock gating can be moved up closer to the root (i.e., the power/ground pad(s)), for more power savings [2].

Operand isolation is the technique that keeps the inputs of the data-path operators stable whenever the output is not used. Special circuitry is required to identify redundant computations of data-path components and to prevent unnecessary switching activity. Both CG and OI techniques can be implemented automatically with standard tools such as Synopsys Power Compiler [2], or manually, by inserting necessary circuits at the RTL.

Reducing the physical load capacitance is a technique to reduce dynamic power consumption. Low-level design parameters, such as size and wire length of transistors, decide this physical capacitance. We can reduce this capacitance by decreasing transistor sizes or by decreasing wire length and/or width at the cost of performance degradation.

Dynamic Frequency Scaling (DFS) is a technique which varies the clock frequency during run time. This technique can reduce dynamic power consumption linearly (Eq. 27.2). However, this also degrades overall performance and does not save total energy consumption. Thus, we can use this technique when reducing peak or average power dissipation, when reducing the temperature of the chip is the major concern.

Lowering the supply voltage is a very attractive method of power saving, because it reduces dynamic power quadratically (Eq. 27.2), while reducing leakage power (Eq. 27.4). However, this technique also increases the delay of CMOS gates inversely. Thus, logical and architectural compensation is necessary for this degradation of performance. The technique of scaling the supply voltage during run time is called *Dynamic Voltage Scaling (DVS)*. However, since reducing the voltage increases gate delays, we also have to reduce the clock frequency for proper operation of the circuit. *DVS* is therefore commonly used in conjunction with *DFS*.

On the other hand, there are techniques that primarily decrease the subthreshold leakage power.

Multiple threshold voltages (V_t) provide a trade-off for leakage power and speed. The high-V_t transistor has a leakage current that is roughly one order of magnitude lower than that of the low-V_t transistor, at the cost of reduction in performance. Thus, the low-V_t transistors are preferred for use in timing critical paths, whereas the high-V_t transistors are used for the rest of the paths. According to Eq. 27.6, this technique exponentially decreases the subthreshold leakage. However, increasing the threshold voltage can decrease logic performance as well, as described in the equation below :

$$f \propto \frac{(V_{dd} - V_t)^{\alpha}}{V_{dd}} \tag{27.8}$$

where f is a frequency, V_{dd} is the supply voltage, V_t is the threshold voltage, and α is a constant.

Decreasing the size of circuits can reduce leakage power. This is because the total leakage current is proportional to the leakages that are consumed in all transistors in a circuit. Minimizing cache size and reducing unnecessary logic in the chip will be helpful in reducing the actual number of transistors and the corresponding leakage power. However, this is not always possible, because reduced logic may degrade performance.

Power gating with *sleep transistors* is reducing the count of the active transistors dynamically, by blocking the power supply to the idle portion of circuits. Problems with this method might include difficulty in predicting the exact time and portion of the idle part of various components, and minimizing the overhead for this by turning them off or on.

Cooling the system is also helpful in reducing leakage power. Various cooling techniques such as blowing cold air, refrigerating the system, or even circulating costly liquid nitrogen have been proposed and used for several decades. This technique has several advantages, such as decreasing subthreshold leakage power significantly and preventing degradation of reliability and lifecycle of a chip. This technique also increases the overall speed of chips, because electricity has smaller resistance at lower temperature. In spite of the aforementioned advantages, cost and cooling system power consumption are major limitations to applying cooling technology to every chip [87].

Again, only the most popular and widely used power minimization techniques have been presented in the section.

27.2.3 Overview of System-Level Power Estimation Methodologies

There have been several power estimation methodologies for specific components in an SoC such as processors [19,27,48,64,89,94,98], various communication fabrics [36, 83–85] and memories [67], and developed power examination tools such as SimplePower [109] and Wattch [19]. There are relatively few methodologies for customized ASIC blocks, due to their extreme heterogeneity. Few researchers have tried to make comprehensive power models for all these components [11, 58, 78]. These models still simplify the power model for a specific component (e.g., the two state processor power model for PowerViP [58]).

With the conventional approach, designers need power estimation models at each of the design abstraction levels, in order to guide design decisions that affect power dissipation. Existing power estimation techniques create power models that map onto, and are useful only at, a particular level. For instance, a technique that is readily applicable at the detailed functional level cannot be easily used at the higher functional level, which is unaware of the detail functionality of the design. Furthermore, if this technique is used at the lower levels, it fails to exploit the additional accuracy in the control and data paths and suffers from an abstraction mismatch. Similarly, cycle-accurate power estimation tools are applicable to the detailed microarchitectural level of the ESL design flow, but cannot be easily ported to higher-level architectural models that lack microarchitectural detail such as the methods described in ▸ Chaps. 25, “Hardware-Aware Compilation” and ▸ 26, “Memory-Aware Optimization of Embedded Software for Multiple Objectives”. The mismatch between power model granularity and level of detail captured at an ESL design level thus limits the applicability of current power estimation techniques across an ESL flow.

In [81], a comprehensive multi-granularity power model generation methodology that spans the entire ESL design flow (Fig. 27.5) was reported. Using industry-standard design flows (Fig. 27.6), this methodology can quickly generate multiple power models ranging from the simplest two-level, coarse-grained model for early power estimation, to the most accurate cycle-accurate model (Fig. 27.8) that allows designers to explore the impact of using power optimizations with minimal manual interference and effort. Our proposed approach is based on the concept of hierarchical decomposition. This decomposition is aided by a tripartite hypergraph model of processor power that can be iteratively refined to create power estimation models with better accuracy (Fig. 27.7). The methodology serves a vital function in supplying a designer with multiple derivative processor power estimation models that match the increasing accuracy of the design, as it is successively refined from the functional, to the architectural and then down to the cycle-accurate microarchitectural stages in an ESL design flow (Figs. 27.7 and 27.8). The feasibility of this approach was demonstrated on an OpenRISC and MIPS processor case study, and present results to show how multi-granularity power models generated for the processors provide designers with the flexibility to trade-off estimation accuracy and simulation effort during system-level exploration.

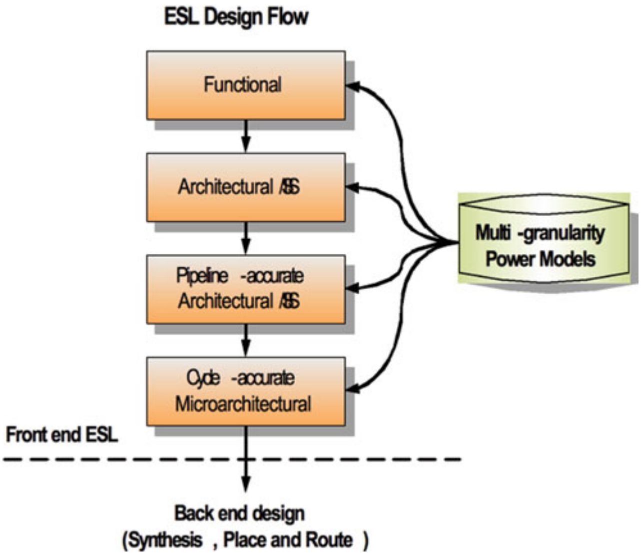

ESL Design Flow
Functional
Architectural ASS
Pipeline -accurate Architectural ASS
Cycle -accurate Microarchitectural
Multi -granularity Power Models
Front end ESL
Back end design (Synthesis , Place and Route)

Fig. 27.5 ESL design flow for embedded processors

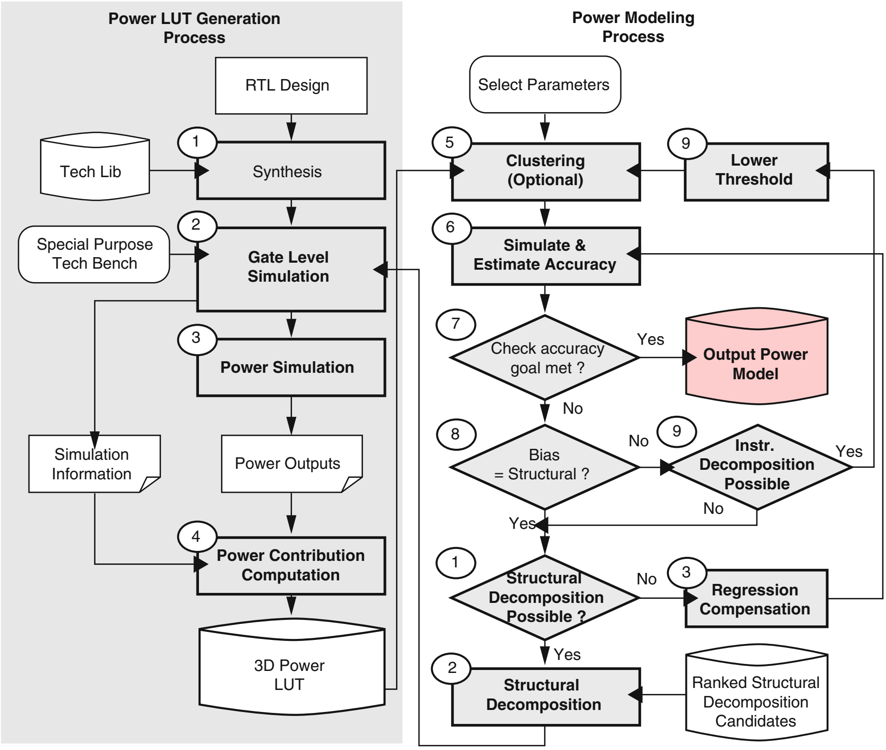


Fig. 27.6 Power model generation methodology

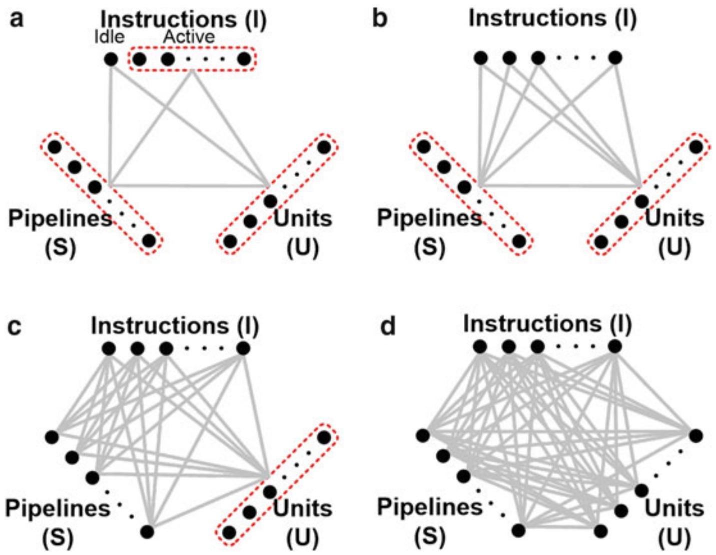


Fig. 27.7 Tripartite hyper-graph H(P), (**a**) simplest two-state power model, (**b**) power model with set I decomposed, (**c**) power model with sets I, S decomposed, (**d**) power model with sets I, S, U decomposed

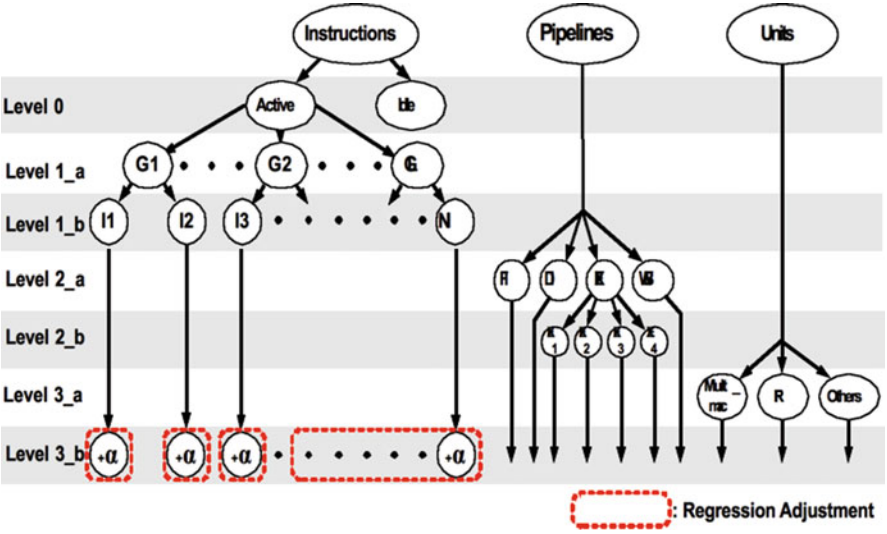


Fig. 27.8 Hierarchical power model for OpenRISC processor

Figure 27.9 shows the average absolute cycle error (E_{AAC}) and relative estimation effort in terms of simulation overhead, for the generated power models for OpenRISC. The power model at Level 0 has a large error of over 20%, which subsequently reduces for the more detailed power models. The Level 3_b power

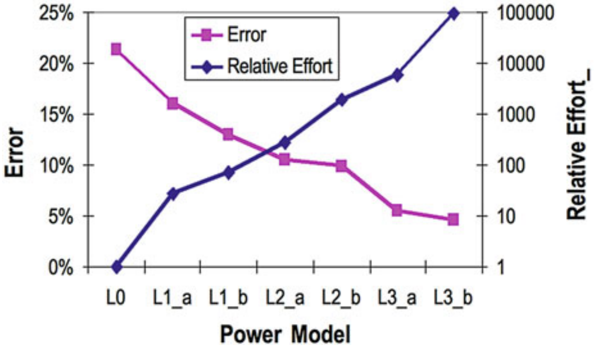


Fig. 27.9 Average absolute cycle error and relative effort for power models

model has an approximately 5% error, which is extremely good compared to gate level estimates. The error in such a detailed model occurs because of several factors, such as the inability to capture the layout and consequently accurately model intra-processor interconnect length, and wire switching.

Figure 27.10 shows a comparison between system-level and gate-level normalized power for the "mul" testbench executing on OpenRISC, across different ESL design flow levels. The figure shows how the coarse-grained *Level 1_b* instruction-set model at the architectural/ISS level is unable to track the power variation very accurately due to the absence of a pipeline at that level. When the pipeline is captured, as in the *Level 2_b* case, then accuracy improves slightly. However, it requires a more detailed *Level 3_b* model which additionally captures the structural units in a cycle-accurate manner, to accurately track the peaks of the gate level power waveform. The power estimated at this level can allow designers to accurately estimate peak power of the processor at simulation speeds that are $100 - 1000\times$ faster than gate-level power simulation. Such a model is extremely useful for determining the thermal and electrical limits of the design and can guide the selection of the appropriate packaging to prevent hotspots and thermal runaway.

27.2.4 Cache Power Modeling

Even though the processor power model described above deeply investigated only the power of core components, we can take account of the cache power for the more realistic embedded processor power model with equation below:

$$P_{processor} = P_{core} + P_{cache} \tag{27.9}$$

where $P_{processor}$ is the cycle-accurate power for the entire processor, P_{core} is cycle-accurate power of the core (which is available from the our methodology), and P_{cache} is the cycle-accurate power for the cache component (which may be obtained from the available memory tool such CACTI [42]).

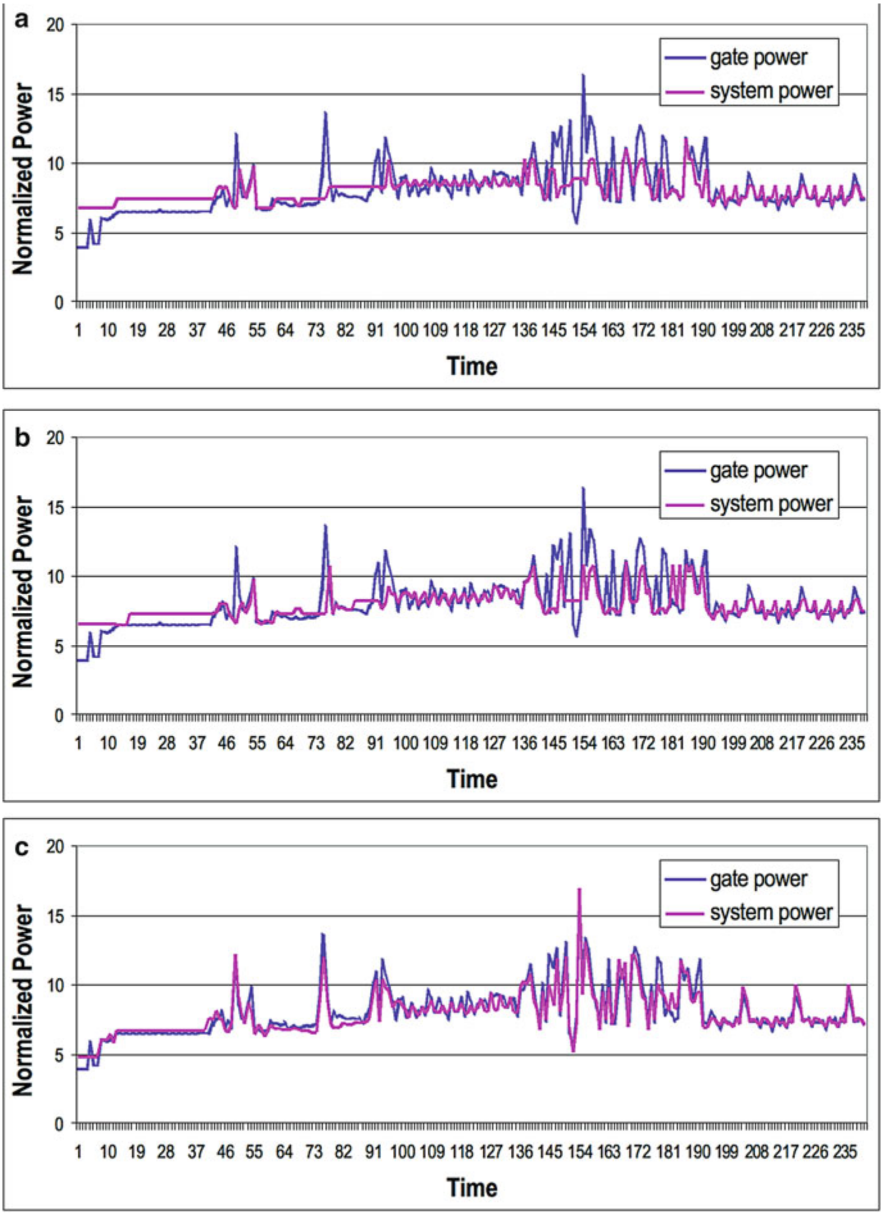


Fig. 27.10 Relative power waveform comparison for the "*mul*" testbench on OpenRISC (Unit for Time: 20 ns). (**a**) Level 1_b. (**b**) Level 2_b. (**c**) Level 3_b

If we are interested in simple average power consumption of a processor, the power model can be formulated as:

$$P_{processor\_avg} = P_{core\_avg} + \frac{nP_{cache\_hit}}{N} + \frac{mP_{cache\_miss}}{N} \tag{27.10}$$

where $P_{processor\_avg}$ is the average power value for the entire processor, P_{core_avg} is the average power value of the core, P_{cache_hit} is the average power value of cache hit (access power), P_{cache_miss} is the average power value of cache miss (idle power), n is total cache hit time during the execution, m is the total cache miss time during the execution, and N is the total execution time.

Rodriguez et al. [92] investigated power consumption of cache with varying sizes (from 16 to 256 K) and associativities (from 1-way to 16-way) for the 65 and 32 nm technology libraries as shown in Fig. 27.11. In fact, current and future SoC designs will be dominated by embedded memory as projected by the ITRS reports which indicate that memories will continue to be a major fraction of any SoC in terms of both area and power [46].

27.3 Thermal and Reliability Issues and Modeling in the Nano-CMOS Era

Downscaling of chips or the continued shrinkage in gate length has naturally increased the power density of chips. Resulting high temperature of chips became one of the biggest issues in chip design, and those thermal issues are becoming more problematic with aggressive technology scaling. In extreme cases, some parts of a chip can be burned out leading to chip failure in the end; thermal runaway, which is caused by positive feedback between increased leakage current and high temperature, can be thought of as one example of such a case. In addition, as we put a lot of heterogeneous components on a chip, the thermal distribution of a chip tend to become nonuniform, i.e., some parts of a chip are hotter than the others due to different processing tasks in different parts of a chip. Implementing multiple cores instead of increasing the clock frequency of a single core became a trend in processor design as a way of alleviating the burden of high power consumption and enormous heat generation [38], and this trend also plays a role in making the thermal distribution nonuniform over a chip to some extent. Especially when thread mapping among the multiple cores is not well-balanced, nonuniform thermal distribution can become a lot worse, resulting in multiple localized temperature maxima, which are usually termed hotspots [38]. According to [14, 15, 103], temperature within a chip can vary as much as 50 ℃ across a die, and examples of this nonuniform thermal distribution are given in Fig. 27.12.

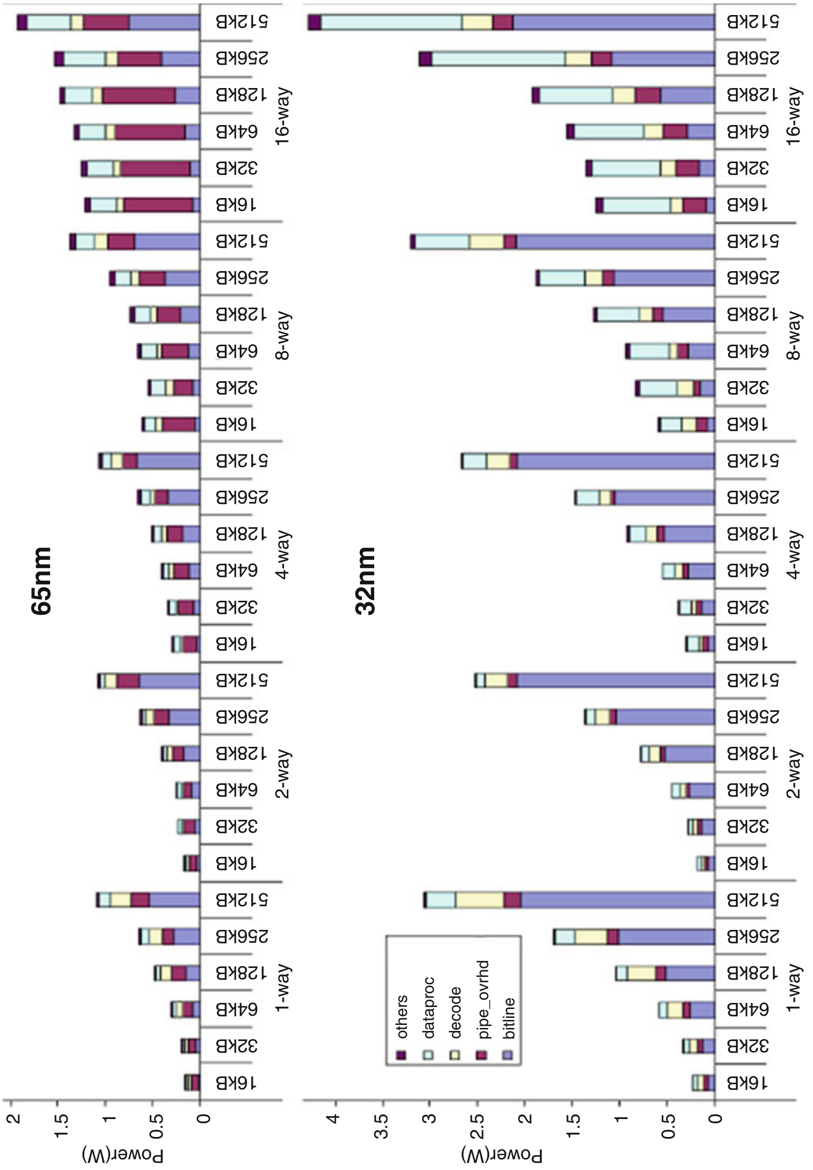


Fig. 27.11 Power breakdown for the 65 nm and 32 nm libraries varying cache sizes and associativities [106]

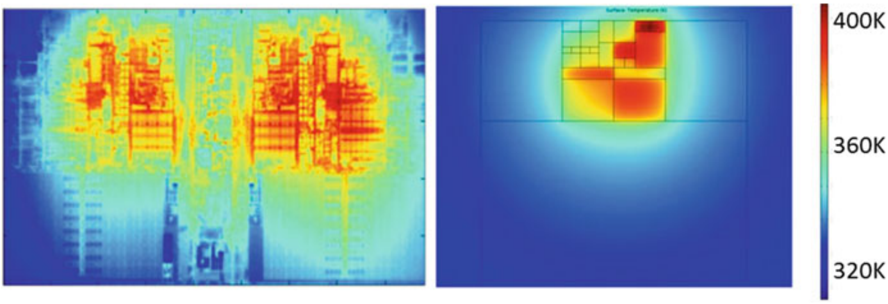


Fig. 27.12 Examples of nonuniform thermal profiles [77, 111]

Hotspots and thermal gradient may result in various kinds of issues: reduced reliability of a chip due to electromigration, [14] timing failure or communication error between functional blocks in a chip due to increased clock skews, higher cost than before for cooling solutions such as heavy cooling fans, heat sinks, etc. [103]

27.3.1 Reliability

One of the serious issues that can be caused by high operating temperatures and a nonuniform thermal distribution over a die is the reduction in the reliability of interconnects and the resulting short life expectancy of a chip due to electromigration [14]. Electromigration is the result of momentum transfer from the collision between electrons and the atoms forming the lattice of the material, and it can cause void or hillock formation along the metal lines in extreme cases. With CMOS technology scaling, the reliability and the life expectancy of interconnects in a chip are becoming more susceptible to electromigration than before. Black's equation or its modified equation [15] given below have been widely used as a way of modeling and predicting the Mean Time to Failure (MTF) of interconnects subjected to electromigration:

$$MTF = \frac{A}{J^n} e^{\frac{E}{kT}} \tag{27.11}$$

In this equation, A is a constant that is determined by the material properties and the geometry of the interconnects, J is the current density, n is a scaling factor that is to be determined experimentally, E is the thermal activation energy depending on the used material, k is the Boltzmann's constant, and T is the absolute temperature of the metal in the unit of kelvin. The current density exponent n is usually set to a value between one and two, and it depends on the failure mechanism [79]; a value close to one characterizes well the failure due to void growth [62]; a value close to two represents the failure due to void nucleation quite well [100]. In the equation, two dominant factors determining the MTF of interconnects are the current density

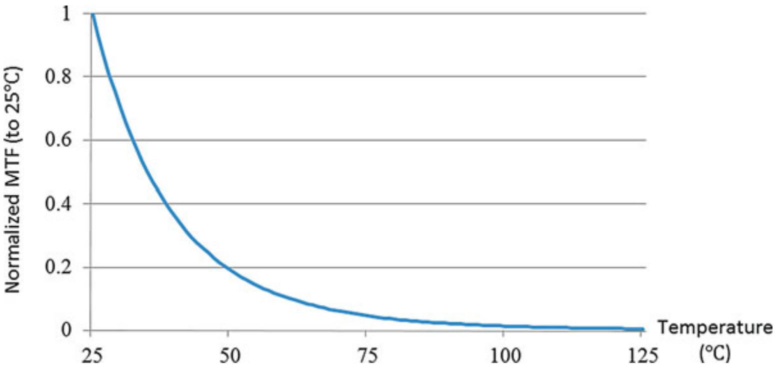


Fig. 27.13 Trend in MTF as a function of temperature

J and the temperature *T*. As CMOS technology scales down, the current density of interconnects generally increases [60], so the life expectancy of interconnects will decrease. To make it worse, the MTF decreases exponentially with respect to the temperature of interconnects. For example, when the temperature of an interconnect changes from 45 to 65 ℃, the life expectancy of the interconnect is reduced by 70% roughly, and the chip will fail much sooner than before if we design chips in a traditional way without proper consideration on thermal issues and adequate cooling solutions. The trend in MTF, which is normalized so that the MTF at 25 ℃ is to be one, is given in Fig. 27.13 as a function of temperature.

As process scaling develops further, the top metal layers get closer to the substrates, and this will further intensify the impact of thermal gradients of substrates on the thermal profile of interconnects [45]; thus, the reliability or the MTF of interconnects decreases exponentially with the increase in the temperature of substrates. In order to improve the reliability or the MTF of interconnects, it becomes indispensable to manage the thermal distribution of a chip dynamically and also to consider the thermal distribution of substrates during chip design or interconnect design stage so that we can avoid hot regions or hotspots on the substrates for the routing.

27.3.2 Dynamic Thermal Management

As we discussed in previous sections, temperature plays a critical role in the reliability, the performance, and the power consumption of a chip in current and future CMOS technology nodes. Therefore, temperature of a chip, especially in case of a high performance chip, should be managed in a smart way at run time so that the maximum temperature can be controlled and also temperature can be evenly distributed both temporally and spatially for better reliability and performance of a chip. According to [39], cost for the implementation of cooling and packaging solutions was expected to increase at an alarming rate with the thermal dissipation of

65 W or higher; hence, thermal management of a high performance chip is also quite crucial in terms of cooling and packaging cost. A large number of techniques for Dynamic Thermal Management (DTM) have been proposed and developed in recent years as ways of limiting the peak temperature of a chip or managing the temporal and spatial temperature variation of a chip through proper resource management [18, 53]. Those techniques can be roughly classified into one of two categories based on how the source management is performed: hardware-based techniques and software-based techniques.

Hardware-based DTM

The relationship between temperature and power dissipation is quite complicated, but temperature can be managed to a certain extent by controlling power consumption of a chip. One of the simple power management techniques, which is called clock gating, began to be used generally in the early 2000s [39]; dynamic power consumption can be minimized by disabling the clocks in a functional block when the functional block is not in use or when the temperature of the functional block reaches a threshold. Clock gating is relatively simple to implement and has good cooling capability because we can effectively reduce the power consumption of a clock tree, which may consume up to around 70% of total dynamic power [37], but the performance degradation is quite high.

Changing dynamically the supply voltage and the clock frequency of a processor based on the workload can be effective in reducing the dynamic power consumption because of the quadratic relationship between dynamic power and the supply voltage, and this technique is called Dynamic Voltage and Frequency Scaling (DVFS) [102]. In case of a processor consisting of multiple cores, the supply voltage and the clock frequency settings of each core can be scaled independently, and it is termed local DVFS or distributed DVFS or per-core DVFS [26], while the chip-level voltage and frequency control is usually termed global DVFS [37]. Additional hardware components and increased design complexity to support multiple clock domains or multiple Voltage/Frequency Islands (VFIs) might become a critical issue especially in case of processors with a large number of cores [40].

Fetch gating [10, 102] is another way to cool down a chip through power consumption reduction; it controls the instruction activity in the pipeline by throttling the fetch stage, and its performance on power reduction and thermal management highly depends on the implemented throttling mechanisms as expected.

Software-based DTM

A simple temperature-aware task scheduling technique for single-threaded processors was proposed in [93]; kernel monitors the CPU activity of each process and the temperature readings from a thermal sensor. When the temperature of a chip becomes higher than a threshold, the kernel identifies processes that use more CPU activities than a predefined value, and then slows them down for cooling purpose. Even though it was simple, it worked effectively to some extent. This basic idea was extended to temperature-aware scheduling techniques for processors that support multi-threading or have multiple cores. For example, a temperature-aware

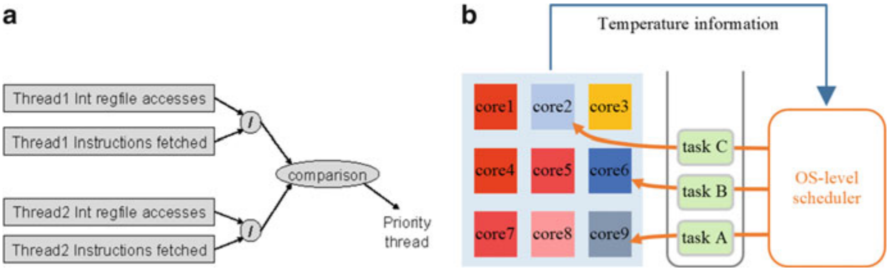


Fig. 27.14 (**a**) Thread selection when the integer register file is thermally critical [32], (**b**) Thermal aware task scheduling for MPSoC [29]

scheduling technique for simultaneously multi-threading (SMT) processors was proposed in [32]; it manages the execution of threads selectively and dynamically based on the probability of heat generation of each thread, and hardware event counters [56] are used for the estimation of the heat generation probability. In [29], a scheduling method specifically targeting Multi-Processor Systems-on-Chips (MPSoCs) was proposed; for each core or processor, the probability of workload assignment is calculated and updated regularly based on the temperature history in the past, and one core with the highest probability is selected when a new workload assignment is required.

When there are multiple cores or processors in a chip, process or task migration can be used effectively in order to balance the thermal distribution among all cores and also to improve the performance as a result; in [33] , a task migration technique was used on top of local DVFS, and it successfully avoided all thermal emergencies, and also achieved 2.6 times speedup when compared with the base case of using local clock gating without task migration. Figure 27.14 illustrates such systems.

27.3.3 Thermal Sensors

As discussed in previous sections, DTM solutions use temperature information to manage the thermal distribution of a chip. Performance Counter-based temperature information can be used for thermal management [56], but the information is not a direct representation of thermal behaviors of a chip most of the time, and it can supply approximation at best. In that sense, it is far better to use the temperature information from thermal sensors because it represents actual thermal behavior of a chip. Each thermal sensor basically provides point-wise temperature information. Thus, it would be better to use a large number of thermal sensors in order to have correct temperature information at any locations of interest on a chip. As for the locations of interest, hotspots need to be monitored first for better reliability and performance, and also for the reduction in power consumption of a chip just as we discussed in previous sections. In addition, a lot more thermal sensors need to be deployed across a die so that the thermal distribution over a die can be monitored and balanced out for the increased reliability of a chip and also for the prevention of performance degradation. However, it is not reasonable to allocate as many thermal sensors as possible on a small-sized chip in reality due to a lot of practical design constraints [63] power consumption and heat generation of thermal sensors, routing and placement issues, etc. As a result, quite a large number of methods have been proposed regarding how to select the number of thermal sensors properly and how to allocate them on a die in order to have accurate temperature readings at any locations of interest on a die at run time.

Another issue to be resolved is the accuracy of thermal sensors; a thermal sensor in a 0.35μm 2.5 V digital CMOS technology, which was implemented in a general purpose microprocessor in the late 1990s, had the reading accuracy of ±12 ℃ with a resolution of 4 ℃, [96]. Since then, great improvement has been made in its reading accuracy, and recent sensors report accuracy of ±1 [63], but there still remains a lot of work to be done especially when it comes to the design of on-chip thermal sensors that are fully compatible with digital CMOS technologies.

27.3.4 Sensor Allocation: Hotspot Monitoring

Since the late 1990s and the early 2000s, thermal distribution of a chip has become a lot more complicated due to a large number of hotspots spreading across a die, and multiple thermal sensors came into play to monitor the temperatures of hotspots more efficiently and accurately.

One simple way to place multiple thermal sensors on a die is to place them on a uniform grid. As a result, some hotspots might not be detected, and the accuracy will be quite limited especially when a small number of thermal sensors is used. Linear interpolation technique using the temperature readings of four neighboring thermal sensors was proposed for the estimation of the maximum temperature of a chip [69];

When thermal distribution of a chip is available, this information can be used for sensor allocation, and thermal sensors can be allocated in a smart way so that the hotspots of a chip can be monitored correctly while minimizing the number of thermal sensors. In [72], a sensor allocation algorithm divides the die area into an array of blocks using the information on hotspot locations, and the size of each block is adjusted in such a way that all hotspots in each block can be covered and monitored by a single thermal sensor assigned to the block. This method works well when the number of hotspots is not large, but with the increase in the number of hotspots, a lot more thermal sensors will be required.

A thermal sensor allocation method based on *k*-means clustering [65] was proposed in [71]; each and every hotspot is assigned to one of *k* clusters recursively, where *k* is the number of thermal sensors, so that the Euclidean distance between the centroid of a selected cluster and the hotspot is minimized. Then, *k* thermal sensors are assigned to the centroids of those *k* clusters. However, this method might

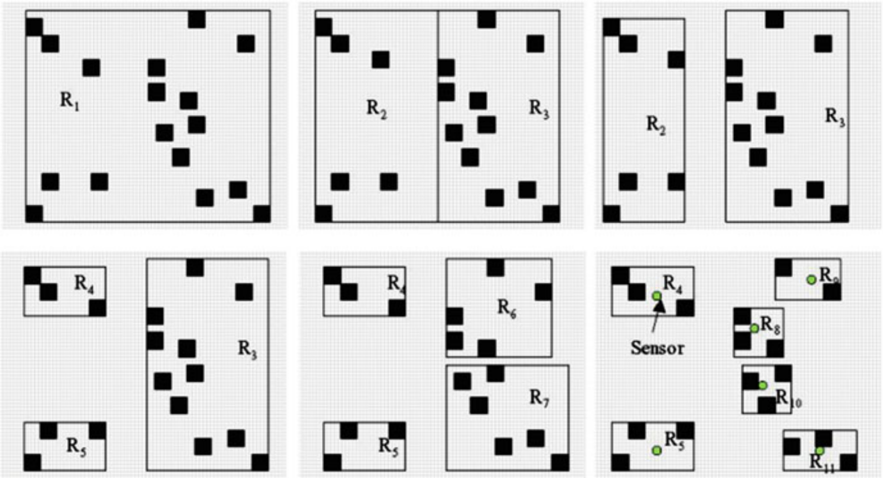


Fig. 27.15 Recursive bisection based thermal sensor allocation [72]

produce some unreasonable results especially when remotely located hotspots have smaller temperature differences than closely located hotspots. Figure 27.15 shows an example of such systems.

27.3.5 Sensor Allocation: Full-Chip Profile Reconstruction

In recent years, a large number of new sensor allocation methods have been proposed to support full-chip thermal profile reconstruction at run time from the temperature readings of a small number of sensors. Sensor allocation is performed with a view to a better run-time thermal profile reconstruction from the beginning, and the number and the locations of thermal sensors are determined accordingly. Fine-grain DTM solutions can make full use of the detailed temperature information from full-chip profile reconstruction, especially on multi-core processors [88]; task migration among cores can be performed more efficiently, and the thermal behavior and static power consumption of caches, which consume a large portion of the die area, can be optimized [47, 49].

In [77] (Fig. 27.16), energy analysis in frequency domain was used for sensor allocation; the main idea of this method is that thermal sensors should be distributed in proportion to the high-frequency energy in frequency domain so that more sensors can be assigned to regions with large thermal variations. This method alternates vertical bisection and horizontal bisection, and then compares the high-frequency energy of the two bisected regions. Thermal sensors are allocated proportionately, and the bisection continues until all thermal sensors are assigned.

In [112], a statistical methodology was developed for sensor allocation and full-chip thermal profile reconstruction; the entire die area was divided into a 16-by-16

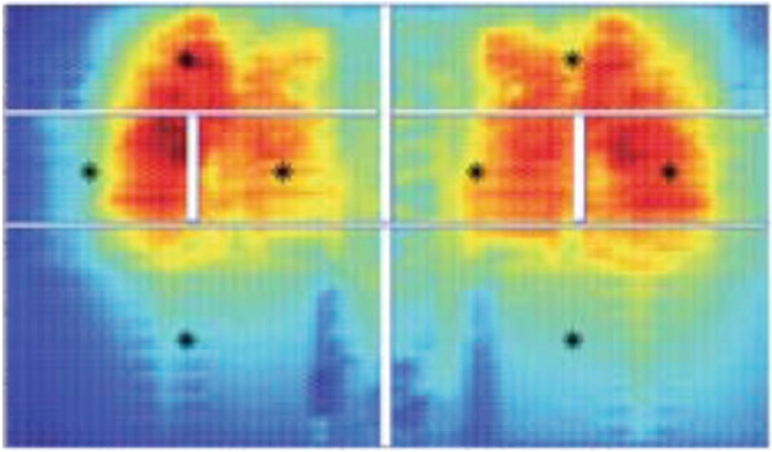


Fig. 27.16 Energy-aware thermal sensor allocation [77]

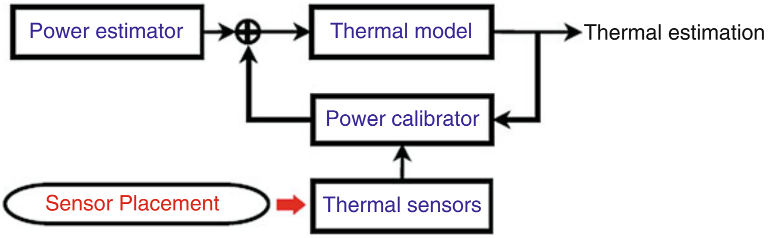


Fig. 27.17 Thermal profile estimation based on sensor-assisted power estimation [107]

grid, and a set of nodes on the grid were selected so that the thermal correlation among them can be minimized, and the thermal correlation between the selected nodes in the set and the nodes outside the set can be maximized at the same time. In this way, each thermal sensor can provide as much temperature information as possible on the non-sensor nodes, while the redundancy among the sensor nodes is minimized.

One way to have accurate temperature information of a chip is to solve the heat differential equation directly with correct power information [43]. Performance counter-based run-time power estimators [86, 108] can be used to supply power information at run time, but they tend to have some power estimation errors. A new approach to achieve good temperature estimation based on the differential equation was proposed in [107] (Fig. 27.17), and it exploits the temperature readings of thermal sensors to correct the power estimation errors. According to the simulation results on a dual-core processor and SPEC2000 benchmark suites [3], it achieved the maximum error of 1.2 ℃, and the averaged error of 0.085 ℃ with six thermal sensors.

In [101] a novel approach of using multiple virtual thermal sensors to increase the accuracy of temperature readings was presented; the virtual thermal sensors are generated from a small low-power physical thermal sensor by adaptively switching its calibration points on the run. Simulation results show that the RMS error of temperature readings can be reduced by up to 91.1% with the use of four virtual thermal sensors as compared with a single thermal sensor case.

27.4 Reliability Modeling

Modern highly scaled CMOS circuits suffer from performance and power losses due to short channel effects that exacerbate process variations. Process-induced variations are typically classified as either systematic or random variations. Systematic variations are predictable in nature and depend on deterministic factors such as layout and surrounding topological environment [74]. These types of errors are handled by static redundancy techniques. Random variations, on the other hand, pose one of the major challenges in circuit design in the nanometer regime [12]. This variation shifts the process parameters of different transistors in a die in different directions, which can result in significant mismatch between neighboring transistors [66]. This phenomenon is typically referred to as Random Dopant Fluctuations (RDF). RDF has the dominant impact on the transistors strength mismatch and is the most noticeable type of intra-die variation that can lead to cell instability and failure. These failures are manifested as either an increase in the cell access time or unstable read and write operations. Typically, RDF effects are countered by increasing the operational supply voltage, thus effectively masking away any variations in the individual transistor threshold voltage. Clearly, this leads to higher power consumption. One major aspect of RDF is that the randomness of the variations results in a random distribution of the access errors across the two-dimensional area of a memory [12, 66]. This phenomenon is a key element that can be exploited by cross layer approaches, since random errors are much easier to handle at higher layers of the system via error correction techniques such as data redundancy (coding) or spatial and temporal filtering. Several research efforts considered exploiting this phenomenon by reducing the supply voltage (i.e., Voltage Over Scaling (VOS)) [6, 44, 73, 75, 76, 99], while informing higher network layers of the anticipated increase in memory fault rates.

27.4.1 Memory

Figure 27.18 shows the typical six-transistor cell used for CMOS Static Random-Access Memories (6T SRAM). The cell consists of two cross-coupled CMOS inverters (NL,PL and NR,PR) that store one bit of information, and two N-type transistors (SL and SR) that connect the cell to the bitlines (BLC and BLT). Classically [66] failures in memory cells are categorized as either of a transient nature dependent on operating conditions or of a fixed nature due to manufacturing

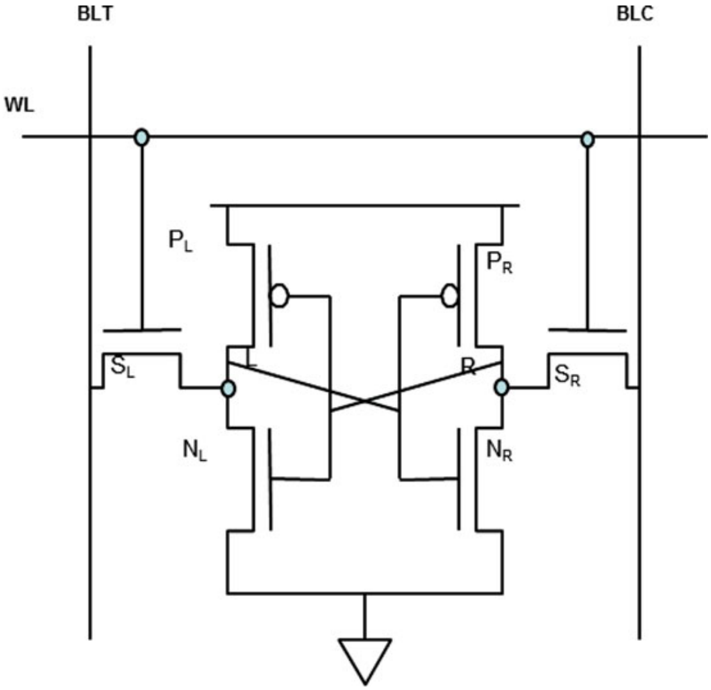


Fig. 27.18 6T SRAM Cell

errors. Symptoms of these failures are expressed as (1) increase in cell access time, (2) write time, or (3) unstable read operations. Conventionally, we assumed that fixed errors are predominant, with a minority of the errors introduced due to transient effects. In sub-100 nm designs, RDF has the dominant impact on the transistors strength mismatch and is the most noticeable type of intra-die variation that can lead to cell instability and failure in embedded memories. RDF has a detrimental effect on transistors that are colocated within one cell, by creating a mismatch in their intrinsic threshold voltage Vt. Furthermore, these effects are a strong function of the operating conditions (voltage, frequency, temperature, etc.).

The total cell failure probabilities for different $\mathbf{V}_{dd}$ as a function of T_{Max}, $P_T[Fail] = Prob[RAF \cup WF \cup DRF]$, are shown in Fig. 27.19 where RAF is read access failure, WF is write failure, and DRF is destructive read failure [31]. This figure illustrates that designers can trade off $\mathbf{V}_{dd}$, performance and error (failure) tolerance to achieve an optimal solution for a given set of conditions. It is important to make a distinction between errors and performance. Performance here is taken to mean achieving a specific ($\mathbf{T}_{Max}$) target as a predefined speed for the SRAM cell, while errors are taken to be hardware malfunction such as **RAF**, **WF**, etc. Intuitively, for a specific performance target, the designers can trade off error tolerance versus supply voltage. In other words, to achieve a low power solution, the designer must first decide on the acceptable level of error tolerance that is

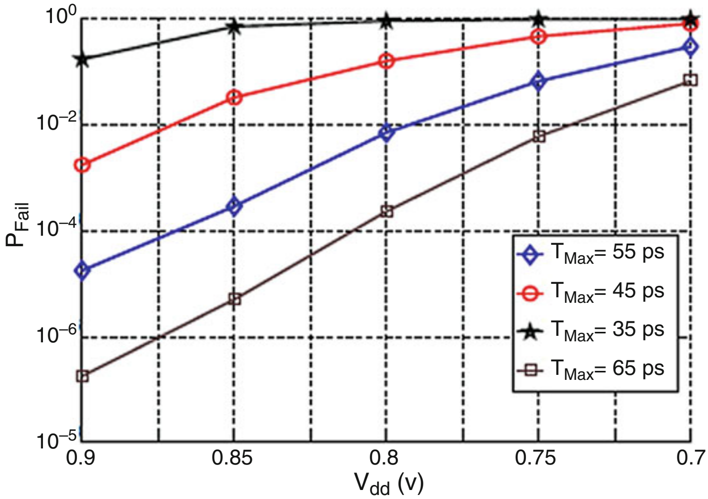


Fig. 27.19 Total cell failure probabilities

permissible by the application and the overall system design while still maintaining the required performance. Given that level, and a required performance level (i.e., $\mathbf{T}_{Max}$), the designer can select the appropriate $\mathbf{V}_{dd}$ from Fig. 27.19. For instance, consider the case that a wireless receiver using a Turbo Decoder is working at nominal $\mathbf{V}dd = 0.9\,v$ and at the failure rate of 10^{-7} and delay of $\mathbf{T}_{Max} = 65\,ps$. It has been shown in [50] that this system can handle memory errors up to 0.1% (10^{-3}). From Fig. 27.19, one can find out that by dropping the $\mathbf{V}_{dd}$ from 0.9 v to almost 0.775 v, the error is still less than 10^{-3} and the system can work at the same performance level, but at lower power.

27.4.2 Combinational Logic

Unlike memory, the propagation delays (t_{pd}) of arithmetic and logic circuits are highly dependent on the input patterns to the block and its circuit implementation [110]. Therefore, errors are not spatially random and one cannot find a closed form failure model for arithmetic and logic circuits. Applying VOS to logic and arithmetic blocks introduces input-dependent errors (timing violations) at the circuit level. Consider a logic gate Z with two inputs a and b and output x. To characterize this gate, the transistor-level circuit representing gate Z (or extracted from layout for more precise modeling of parasitics) is implemented in a Spice simulation making similar assumptions about threshold voltage, Vth as in Section 27.4.1. A Monte-Carlo simulation is run on the circuit for each of the possible 2^{2n} input-vectors, where n is the number of input signals to the gate, and an input-vector consists of the previous and current states of the inputs. The propagation delays

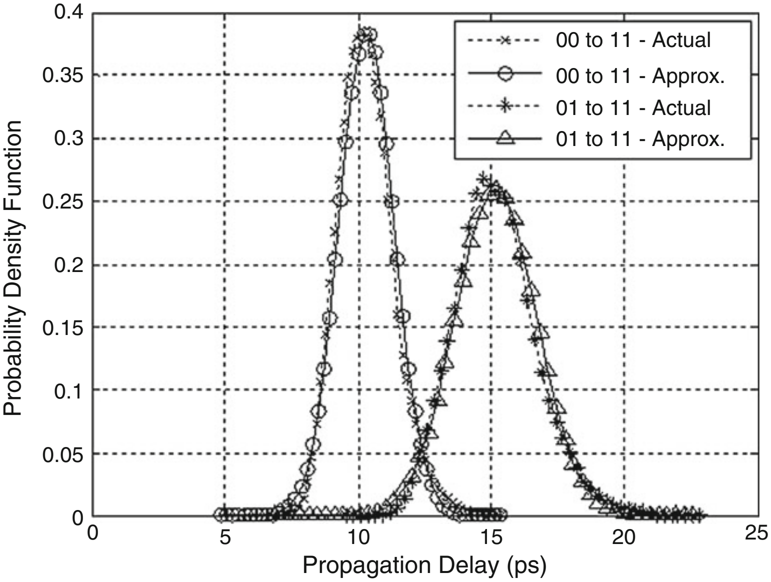


Fig. 27.20 pdf of a 2 input CMOS AND gate

statistics and the average power consumption for each input-vector are measured and stored. Figure 27.20 shows the Probability Density Functions (PDFs) of the propagation delays of a two input CMOS AND gate simulated in a 32 nm process under nominal supply voltage of 0.9 V using predictive transistor-models [1]. Two input-vectors, with input state transitions $ab = 00 \rightarrow 11$ and $ab = 01 \rightarrow 11$, are used. The PDFs of the measured propagation delays for the two scenarios show a very close match as compared a normal distribution approximation $\mathscr{N}\left(\mu_i, \sigma_{\mathrm{i}}^2\right)$. Note that even though the outputs of the two input vectors are the same, their propagation delays are considerably different because of the initial state of the inputs. As circuit size increases, the complexity of modeling such delay distributions quickly becomes unmanageable. To address this challenge, one needs to incorporate circuit-level failures into a system-level simulation. While Statistical Static Timing Analysis (SSTA) [13] rapidly gives useful statistics of propagation delays and timing violations of critical paths, it does not give any information about the specific *input-vectors* that will cause timing violation errors in those paths. Therefore, SSTA cannot be used to address this challenge. On the other hand, Dynamic Timing Analysis (DTA) [61, 106] simulates circuits for functionality to acquire propagation delays on a per input-vector basis. Hence, DTA can be used to address the challenge of trading off reliability versus energy efficiency. In doing so, one can attempt to integrate a circuit simulator (such as Spice) into the system-level simulation to acquire propagation delay results on a per input-vector basis. This, however, will be very costly in terms of processing overhead and simulation time, since the quality and accuracy of DTA is directly proportional to the number of input

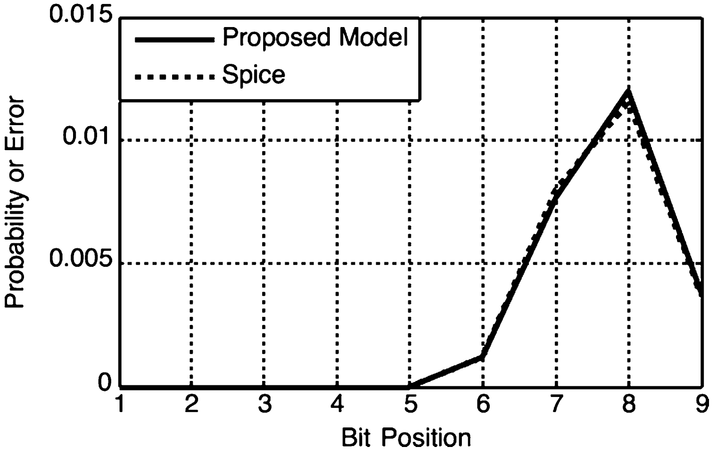


Fig. 27.21 Comparing probability of error per bit from proposed model and from Spice simulation

test vectors used. Simulating a simple digital block for one input-vector in Spice requires run time in the order of few hundred milliseconds. This would be very inefficient for processing large amounts of data. Methods such as [110] attempt to macromodel these distributions and propagate them in a consistent way, allowing the modeling of large combinational components such as Adders, multipliers, and CORDIC. Figure 27.20 compares the probability of error per bit for the adder from the proposed model and from the Spice simulation, and it confirms that the second most significant output bit has the highest probability of error (Fig. 27.21).

27.4.3 Microarchitecture and System Level

In recent years, numerous research efforts have targeted reliability-power-performance trade-offs via software, microarchitectural, and circuit techniques, including several efforts by the Principal Investigators (PIs) as outlined above. For example, in [91] Rinard et al. identify inherent redundancies in computational patterns such as sum and mean calculations and indicate the insignificant impact of resource reduction on such patterns. Moreover, techniques such as varying clock frequencies, skipping tasks, loop perforation, dynamic knobs, and using alternative implementations for key components [7, 9, 41, 70, 90] have also been performed with minimal effect on the accuracy of final performance metrics. Other approaches [24, 104] propose relaxing the correctness at the end results. These approaches allow the user to determine the minimum necessary precision needed at a given point or output of the code and adjust the amount of calculations performed to only satisfy the required accuracy. To use the hardware redundancies, the EnerJ approach [95] introduces an approximated language that enables the developer to distinguish the precise and "approximatable" parts of the code and save energy on portions that can tolerate approximation. Furthermore, a more detailed work by the same research group [110] uses EnerJ as the guarantee for reliability and proposes language constructs that focus more on power-saving possibilities in a pipelined architecture and further specify the hardware implementations of such approximated language. ERSA [59] is a more drastic technique for saving power on a multi-core architecture. It divides the cores to reliable and unreliable cores and proposes to reduce the voltage on all parts of the unreliable cores. Even though there is no error recovery method introduced and the frozen cores are restarted by reliable ones, the output remains more than 90% accurate for the tested set of benchmarks. Earlier works such as the Aura/Odyssey/Coda project [76] investigated mobility and adaptation at the software level via what was termed "application-aware adaptation." Finally, the work of Breuer and Gupta promotes the concept of living with processing errors in some cases in order to improve yield [16, 17].

On the microarchitectural front, researchers have proposed several approaches that attempt to exploit architectural innovations to reduce excessive design margining associated with process variations. Examples include Razor [30, 35, 57] and TEAtime [105] that add extra hardware to correct for errors. The research in [76, 99], and [31] promoted the use of "algorithmic noise tolerance" and proposed using adaptive filters and replication to minimize the impact of scaling V_{dd} beyond the critical region for basic DSP functions, e.g., filtering and other communication blocks. The work proposed in [73] and [75] considers faulty caches and means of dealing with process faults through isolating faulty cache lines. On the circuits front, there has been significant work to achieve low-power operation through a combination of circuit design and technology-dependent optimizations [17, 21–23, 28, 51, 52, 68, 81, 97].

27.5 Interplay between Power, Temperature, Performance, and Reliability

The power, temperature, performance, and reliability of a chip exhibit a complex relationship where a small change in one dimension can potentially affect other characteristics. In order to illustrate the complexity of the interacting metrics or power, performance, and reliability in a dynamically changing environment, we present as an example an embedded memory block. Classically, failures in embedded memory cells are categorized as either of a transient nature (because of operating conditions) or of a fixed nature (due to manufacturing errors). Failures are manifested as (1) increase in cell access time, or (2) unstable read/write operations. In process technologies larger than 100 nm, fixed errors are predominant, with a minority of the errors introduced due to transient effects. This model cannot be sustained as scaling progresses due to the random nature of the fluctuation of dopant atom distributions. In fact, in sub 100 nm design, RDF has the dominant impact on the transistor's strength mismatch and is the most noticeable type of intra-die variation that can lead to cell instability and failure in embedded memories [55].

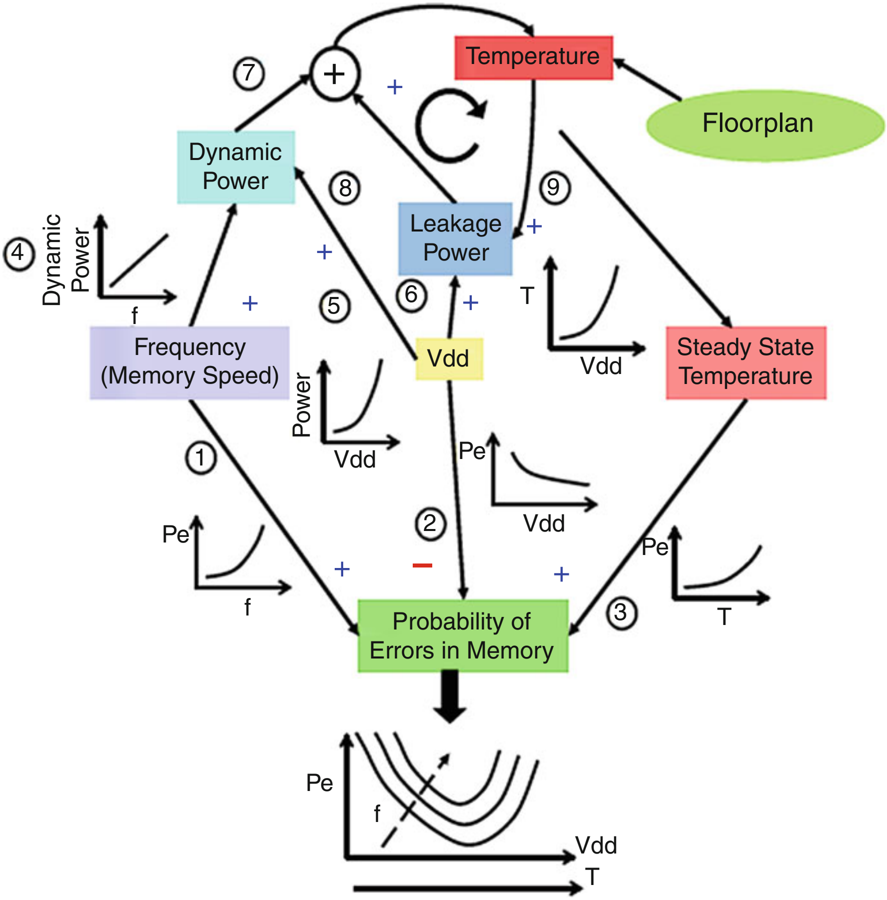


Fig. 27.22 Sensitivity of memory errors to various parameters

RDF has a detrimental effect on transistors that are colocated within one cell by creating a mismatch in their intrinsic threshold voltage, V_t. Furthermore, these effects are a strong function of the operating conditions such as voltage, frequency, temperature, etc.

Figure 27.22 shows how errors in memory are affected by different parameters. As the operating frequency is increased, the probability of memory errors increases (1) because it enforces tighter bounds on the time allowance for memory accesses. Increase in V_{dd} reduces the cell delay and thus causes the errors to decrease (2). The errors in memory increase along with the rise in temperature (3) because of increase in the cell delay. These are not the only relationships that affect memory errors. From Fig. 27.22, we also examine other interrelationships at work: The dynamic power dissipation in memory cells increases with increase in both frequency ($\propto f$) and V_{dd} ($\propto V_{dd}^2$) . The leakage power, on the other hand, increases with V_{dd} ($\propto e^{\beta V_{dd}}, \beta > 1$). Both dynamic power and leakage power determine the operating temperature. Leakage power dissipation of a cell is known to increase superlinearly with increase in temperature. As temperature increases, the leakage power dissipation increases which further elevates the temperature. This "positive feedback loop" between temperature and leakage power stabilizes when steady-state operating temperatures have been reached at which state, all the dynamic and leakage power dissipation is transferred to the environment by the package [30]. This discussion implies that probability of error is not a monotonically decreasing function of supply voltage but rather exhibits a convex behavior as shown in Fig. 27.23. A comprehensive approach to memory/logic design must consider these mutual interdependent relationships. The effect of interplay between V_{dd}, probability of error, and temperature for different cell speeds is shown in Fig. 27.23 where the different curves represent the behavior of memories with maximum allowed times of 70, 67, 65, and 60 ps, respectively. We observe that as the frequency of the cell increases (or the delay decreases), the probability of error also increases. We also observe that an increase in V_{dd} reduces the probability of error but only up to a certain point (marked **X**). After **X**, the rise in temperature due to V_{dd} increases the memory errors. However, for curve we do not observe this behavior because the speed of the cell is low and the probability of failure is not detectable by our simulation setup. In the absence of thermal considerations, these curves would have continued to exhibit decreasing probabilities of error with increasing V_{dd}.

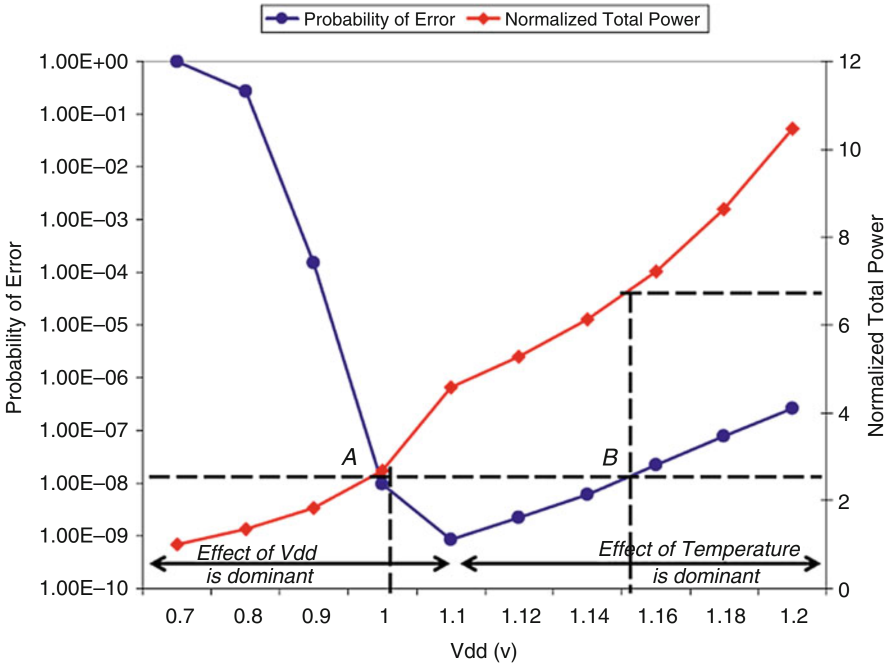


Fig. 27.23 Probability of error for different frequencies

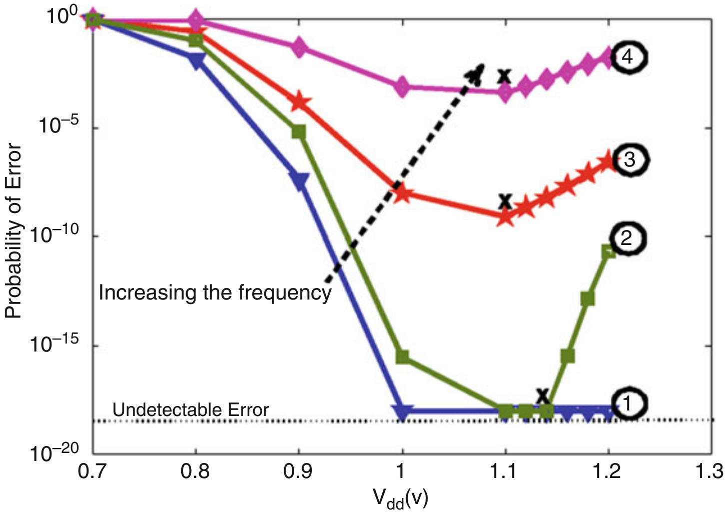


Fig. 27.24 Probability of error and total power

Figure 27.24 shows the normalized total power dissipation and the probability of error for a cell with maximum allowed time of 65ps. Initially, the effect of increase in V_{dd} is dominant and probability of error decreases with increase in V_{dd}. However, at higher V_{dd} the effect of resulting temperature becomes dominant and probability of error increases with increase in V_{dd}. The figure illustrates that for a given probability of failure target, two voltage levels can be chosen that achieve the desired target. For example, at a target probability of error of 10^{-8}, one can select either 1.0 v (Point *A*) or 1.16 v (Point *B*). However, the dynamic power at 1.0 v is 34.5% less than that at 1.16 v because dynamic power $\propto \mathrm{V}_{\mathrm{dd}}^2$. Even without thermal dependence, the leakage power at 1.0 v is 46.1% less than that at 1.16 v because leakage power $\propto e^{\beta \mathrm{V}_{\mathrm{dd}}}, \beta > 1$. Thus, the total power at 1.0 v is 2.5× less than that at 1.16 v. Designers can save significant power by operating at lower V_{dd} voltages while maintaining performance levels.

27.6 Power, Performance, and Resiliency Considerations in SoC Design

While scaling V_{dd} is indeed one of the most effective means of controlling power, it is imperative to understand how other effects can be factored in. For example, incorporating temperature, process variations, etc. will lead to significantly different system policies than those which would be adopted in a non-cross layer aware approach. Figure 27.25 highlights the different phases in error development as V_{dd} is scaled. When V_{dd} is close to its typical value, the system operates normally. As we scale down, errors start to develop. Depending on the system, such errors may

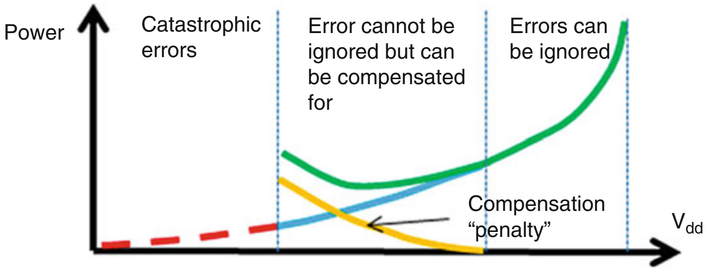


Fig. 27.25 Different phases in error development

be ignored, but only up to a point beyond which it becomes necessary to deploy measures that will compensate, either partially or fully, for the drop in quality. Discussed below, these measures can occur at different layers of abstraction and incur a penalty causing a lessening of the power savings due to V_{dd} scaling. This may lead to a "sweet spot" at which maximum power savings can be achieved. If we keep decreasing V_{dd}, not only would the savings become less, but there typically exists a point at which errors can become so large that the system breaks down and quality cannot be recovered.

How much errors can be tolerated depends on the technology, architecture, and application. Similarly the method of error compensation depends on these parameters as well and can be applied at all the design layers. In the following, we present examples of error tolerance at these various design layers and methods of compensation. These examples serve as data points and are not intended to limit the scope of the discussion.

27.6.1 Architecture-Level Error Tolerance

Consider the case of a stringently error-constrained system. A representative 16 KB processor cache is assumed, with a block size of 16 bytes and associativity of 1, based on an underlying 70 nm CMOS technology. Cache errors can either lead to unstable system behavior or excessive delay due to a cache miss. By varying the voltage of the cache from 0.9 to 0.7 v, it was observed that the miss rate increases from 3.9 to 59.5%, respectively. It is clear that reducing the supply voltage increases parametric errors to an extent where the memory becomes useless. To counter this effect, a modified structure allows the memory to continue operation with minimal impact on the miss rates even at elevated error rates [50]. The proposed architecture is shown in Fig. 27.26 (Red dots indicate parametric errors), where, in addition to the cache block, two other blocks are added, the Bit Lock Block (BLB) and the Inquisitive Defect Cache (IDC). The BLB is an off cache defect map that stores a tag bit to identify faulty locations. These can be updated via a built-in self-test initiated at each configuration update. The IDC acts as a place holder for defective cache words. Due to the random spatial distribution of RDF induced faults, as well

Benchmark	32nm		65nm	
	Min Voltage	Power Saving	Min Voltage	Power Saving
DEC	0.75	43.25%	0.84	10.23%
LISP	0.72	49.35%	0.77	18.36%
MACR	0.75	44.32%	0.83	12.21%
PASC	0.74	45.60%	0.83	11.90%
MUL	0.75	43.12%	0.84	7.93%
SPIC	0.72	44.73%	0.77	15.21%

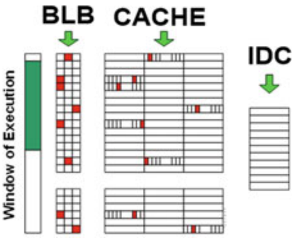


Fig. 27.26 IDC Error-tolerant cache architecture

as the locality of access in a cache, the size of IDC cache could be much smaller than the total size of all the defective words in a cache. Since this cache is masking parametric errors, it can be thought of as a means of tightening the distribution of faults as a function of the change in supply voltage. In this sense, DMS can be used to identify the optimum size of this cache based on an expected distribution of errors due to supply changes. If the IDC size and associativity are chosen properly, the execution window of a process in the cache (after its first pass) will experience very little defective/disabled words.

Both the IDC and the BLB can be assumed to be operated at a higher and safer voltage (within temperature constraints) or utilizing larger devices, while Adaptive Supply Voltage (ASV)/Adaptive Body Bias (ABB) is applied on the cache body. Early simulation results, using standard benchmarks, and a trace-driven simulator are shown in Fig. 27.26 where the miss rate is reduced down to 6.45% (from 59.5% initially) and power savings of more than 40% are reported. The interested reader is referred to [51] for more details about the simulation setup and results. This approach benefits from the spatial randomness of process-induced faults, thus allowing the use of a very small victim cache to perform cache remapping at elevated error rates without reducing the size of the cache.

27.6.2 Application-Level Error Resiliency: Multimedia Applications (H.264)

Consider an H.264 system Fig. 27.27 (left) as a representative application for mobile multimedia systems. One of the biggest challenges is power consumption, which is typically addressed by power management, mainly by reducing the supply voltage. However the range of such a reduction is limited by (1) performance constraints and (2) component reliability under very low V_{dd}.

By design, these systems have built-in error resiliency that has been exploited in many different compression and transmission schemes mainly as a quality trade-off. [54] proposed utilizing aggressive voltage scaling on *embedded memories* resulting in low-power, high-frequency operation, albeit, with errors due to scaling. Based on the error statistics, we propose, analyze, and quantify the performance and overhead (in terms of power and area) of various filtering and mapping techniques

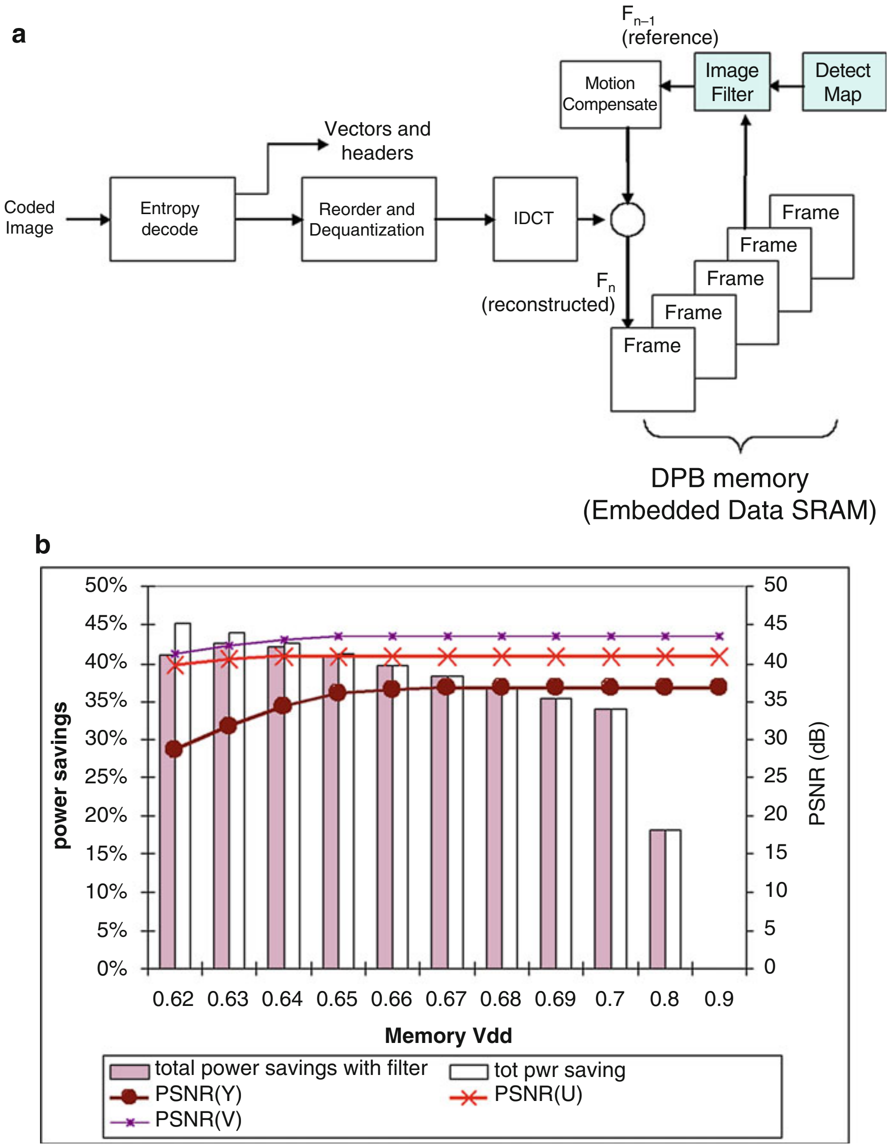


Fig. 27.27 H.264 decoder with filtering (*left*) & Image quality (PSNR) vs power savings at different Vdd levels (Foreman Video) (*right*)

that compensate for the errors, thus enabling the system to operate at lower voltages while meeting system specifications. Finally, we quantify the expected system power savings due to the above mentioned approach. Figure 27.27 (right) shows the results of such an exploration. When we lower V_{dd} on the decoder memories, its reliability decreases and as a result, the output quality drops. This can be compensated for by filtering, which consumes power so the gains from V_{dd} reduction tend to lessen as error rates increase. However, the overall results indicate that good performance (PSNR) can be maintained even at very low V_{dd} while saving over 40% in overall system power consumption. While this case study highlights a significant opportunity in power savings, it requires an important paradigm shift in today's system design flow. Current flow emphasizes compartmentalization between system-level designers and backend (chip) designers, thus necessitating 100% correctness in hardware. The new paradigm de-compartmentalizes this flow and allows system designers to be aware of the physical layer through model abstractions.

27.6.3 Application-Level Error Resiliency: Wireless Modem Application (WCDMA)

In this section, we now extend the discussion to the transmission medium, using Wideband CDMA (WCDMA) as a representative of a wireless physical layer. Figure 27.28 depicts the top-level block diagram of a diversity enabled WCDMA SoC modem [34]. The SoC includes the modem section (RAKE receiver), the coding layer and the protocol layer of the standard. It is based on a dual embedded microcontroller architecture. The symbols from the modem are soft values with 10-bit precision that are available for all the data and control symbols transmitted on the data channels. Naturally, control symbols are very important and thus must be stored in a protected memory with minimum loss. However, data symbols possess a high degree of redundancy typically inserted by the channel coding scheme. Specifically in WCDMA [50], both Turbo and Viterbi schemes are supported. Thus, the data

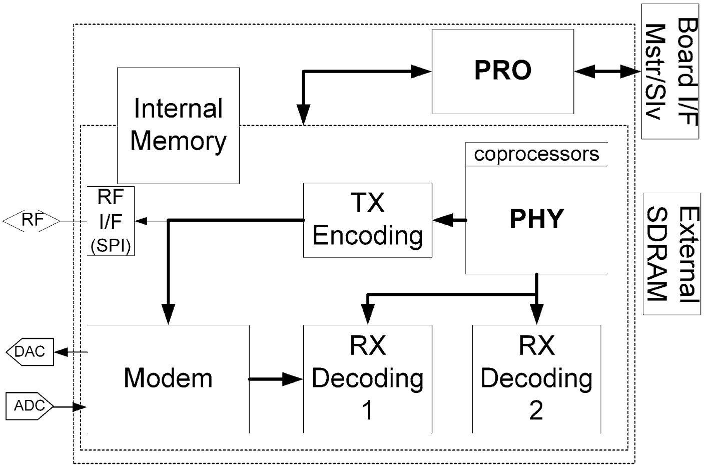


Fig. 27.28 WCDMA chip architecture

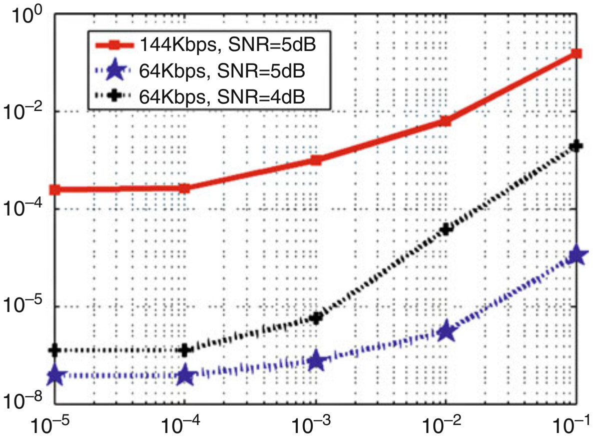


Fig. 27.29 Effect of the WCDMA memory errors on the system Bit Error Rate

memory can be partitioned into defect tolerant and non-defect tolerant sections. A defect tolerant memory is a memory that is used primarily to buffer data and thus can be a target of aggressive power management. It is interesting to note that the data buffering memories (defect tolerant candidates) consume approximately 50% of the overall memory required for the entire modem.

Figure 27.29 shows the effect of the memory errors on the WCDMA Bit Error Rate (BER) for different transmission bit rates. As expected, given the same SNR, for higher bit rates, the memory errors have higher impact on the BER since there is less redundancy in the system. A power analysis of the architecture indicates that the overall memory consumes roughly 45% of the total power. In prior work, the PIs have shown that by applying error aware dynamic voltage scaling a savings of 46% in leakage power and 44% in dynamic power is possible in the error-tolerant memories. It is important to note that these savings are independent of other power-saving methods such as reducing frequency of operation, etc.

27.6.4 Mobile Phone SoC Example

The previous two case studies illustrated the design space exploration for a wireless and a video application. In today's mobile phones, these applications are synergistic and are typically implemented on the same chip. Figure 27.30 illustrates a hypothetical modern mobile phone SoC with a WCDMA physical layer to convey the data stream and an H.264 video codec as the application. While in the previous cases we considered each application separately, in this case, the question arises as to which of the two applications to target for V_{dd} reduction in order to save overall SoC power? To do so we investigated three scenarios: Case (A) Nominal-V_{dd} for the WCDMA modem and aggressive dynamic voltage scaling (AVS) for the H.264

Fig. 27.30 A mobile phone SoC architecture

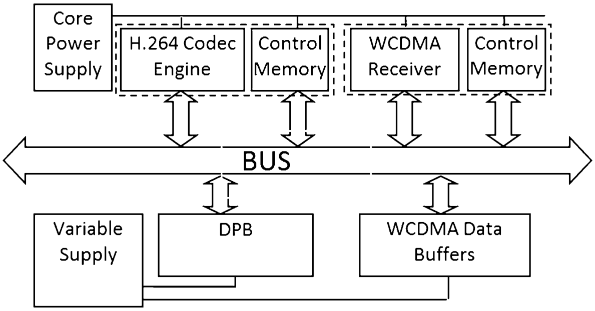
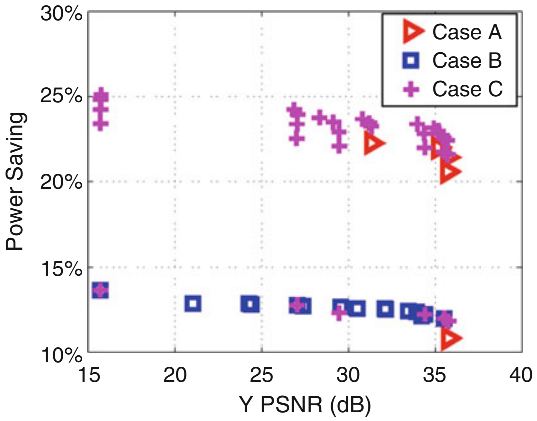


Fig. 27.31 Application-aware design space exploration for the cell phone SoC



decoder engine, Case (B) supply scaling for the WCDMA modem and nominal-V_{dd} for the H.264 decoder, and Case (C) supply scaling for both the H.264 decoder and the WCDMA modem. Based on prior experience, the PIs have with WCDMA systems, it is estimated that the WCDMA modem consumes 72% of the total power whereas the H.264 decoder consumes 28% of the total power in 65 nm technology node [50]. As this portion changes, the gains will scale accordingly. Figure 27.31 illustrates a summary of the results depicting the expected power savings for each case versus the luma (Y) component of the image as a quality metric. From the graph, we observe that some points are inferior to others. In other words, one case may yield higher power savings than another for the same target PSNR. Such points are considered as Pareto-optimal. Overall, Case B appears to be inferior to cases A and C. However, this is a direct result of the ratio of power consumption of the receiver and video decoder. Since the receiver consumes more than 3× the power of the H.264 decoder, one would expect to get more power reduction by supply scaling of the receiver. This situation would be reversed in another system where the ratios are the opposite. As expected, case C yields the most Pareto-optimal design points since it is a superset of cases A and B.

27.7 Summary and Conclusion

This chapter presented a typical design flow for integrating microarchitectural IPBs into complex SoCs that must satisfy performance, power, thermal, and reliability constraints. Toward this end, we first presented different abstraction levels for SoC design that promote IP reuse and which enable fast simulation for early functional validation of the SoC platform. Since SoCs must satisfy a multitude of interrelated constraints, we then presented high-level power, thermal, and reliability models for estimating these constraints. We outlined the complex interrelationship between power, temperature, performance, and reliability of an SoC and illustrated these dependencies. We concluded the chapter with several case studies presenting examples of error tolerance at these various design layers and methods of compensation.

References

1. Predictive technology model(ptm). http://www.eas.asu.edu
2. Synopsys design compiler, primetime px, power compiler. http://www.synopsys.com
3. Standard performance evaluation council, performance evaluation in the new millennium, v.1.1 (2000)
4. Functional Specification for SystemC 2.0. www.systemc.org (2001)
5. International technology roadmap for semiconductors (2011) System drivers. Technical report.
6. Abdallah R, Shanbhag N (2009) Error-resilient low-power Viterbi decoder architectures. IEEE Trans Signal Process 57(12):4906–4917. doi:10.1109/TSP.2009.2026078
7. Ansel J, Chan C, Wong YL, Olszewski M, Zhao Q, Edelman A, Amarasinghe S (2009) Petabricks: a language and compiler for algorithmic choice. In: Proceedings of the 30th ACM SIGPLAN conference on programming language design and implementation, PLDI '09. ACM, New York, pp 38–49. doi:10.1145/1542476.1542481
8. ARM (2001) ARM AMBA specification and multi layer AHB specification, (rev2.0). http://www.arm.com
9. Baek W, Chilimbi TM (2010) Green: a framework for supporting energy-conscious programming using controlled approximation. In: Proceedings of the 31st ACM SIGPLAN conference on programming language design and implementation, PLDI '10. ACM, New York, pp 198–209. doi:10.1145/1806596.1806620
10. Baniasadi A, Moshovos A (2001) Instruction flow-based front end throttling for power-aware high performance processors. In: Proceedings of the 2001 international symposium on Low power electronics and design (ISLPED'01). ACM, New York, pp 16–21. http://dx.doi.org/10.1145/383082.383088
11. Bansal N, Lahiri K, Raghunathan A, Chakradhar S (2005) Power monitors: a framework for system-level power estimation using heterogeneous power models. In: 18th international conference on VLSI design, pp 579–585 doi:10.1109/ICVD.2005.138
12. Bhavnagarwala A, Tang X, Meindl J (2001) The impact of intrinsic device fluctuations on CMOS SRAM cell stability. IEEE J Solid State Circuits 36(4):658–665. doi:10.1109/4.913744
13. Blaauw D, Chopra K, Srivastava A, Scheffer L (2008) Statistical timing analysis: from basic principles to state of the art. IEEE Trans Comput Aided Des Integr Circuits Syst 27(4):589–607. doi:10.1109/TCAD.2007.907047
14. Black J (1969) Electromigration – a brief survey and some recent results. IEEE Trans Electron Devices 16(4):338–347

15. Blair J, Ghate P, Haywood C (1971) Concerning electromigration in thin films. Proc IEEE lett 59:1023–1024
16. Breuer M (2010) Hardware that produces bounded rather than exact results. In: 2010 47th ACM/IEEE design automation conference(DAC), Anaheim, pp 871–876
17. Breuer M, Gupta S, Mak T (2004) Defect and error tolerance in the presence of massive numbers of defects. IEEE Des Test Comput 21(3):216–227. doi:10.1109/MDT.2004.8
18. Brooks D, Martonosi M (2001) Dynamic thermal management for high-performance microprocessors. In: Proceedings of the 7th international symposium on high-performance computer architecture (HPCA'01). IEEE Computer Society, Washington, DC, p 171
19. Brooks D, Tiwari V, Martonosi M (2000) Wattch: a framework for architectural-level power analysis and optimizations. In: Proceedings of the 27th international symposium on computer architecture, Vancouver, pp 83–94
20. Cai L, Gajski D (2003) Transaction level modeling: an overview. In: First IEEE/ACM/IFIP international conference on hardware/software codesign and system synthesis, pp 19–24. doi:10.1109/CODESS.2003.1275250
21. Calhoun B, Chandrakasan A (2004) Standby power reduction using dynamic voltage scaling and canary flip-flop structures. IEEE J Solid State Circuits 39(9):1504–1511. doi:10.1109/JSSC.2004.831432
22. Calhoun B, Daly D, Verma N, Finchelstein D, Wentzloff D, Wang A, Cho S, Chandrakasan A (2005) Design considerations for ultra-low energy wireless microsensor nodes. IEEE Trans Comput 54(6):727–740. doi:10.1109/TC.2005.98
23. Calhoun B, Wang A, Chandrakasan A (2005) Modeling and sizing for minimum energy operation in subthreshold circuits. IEEE J Solid State Circuits 40(9):1778–1786. doi:10.1109/JSSC.2005.852162
24. Carbin M, Kim D, Misailovic S, Rinard MC (2012) Proving acceptability properties of relaxed nondeterministic approximate programs. In: Proceedings of the 33rd ACM SIGPLAN conference on programming language design and implementation, PLDI '12. ACM, New York, pp 169–180. doi:10.1145/2254064.2254086
25. Center for Embedded Computer Systems: SpecC system. http://www.cecs.uci.edu/~specc/
26. Chabloz J, Hemani A (2010) Distributed DVFS using rationally-related frequencies and discrete voltage levels. In: Proceedings of the 16th ACM/IEEE international symposium on Low power electronics and design (ISLPED' 10). ACM, New York, pp 247–252. http://dx.doi.org/10.1145/1840845.1840897
27. Chakrabarti C, Gaitonde D (1999) Instruction level power model of microcontrollers. In: Proceedings of the 1999 IEEE international symposium on circuits and systems, ISCAS '99, vol 1, pp 76–79. doi:10.1109/ISCAS.1999.777809
28. Cho S, Chadrakasan A (2004) A 6.5-ghz energy-efficient BFSK modulator for wireless sensor applications. IEEE J Solid State Circuits 39(5):731–739. doi:10.1109/JSSC.2004.826314
29. Coskun AK, Rosing TS, Whisnant K (2007) Temperature aware task scheduling in MPSoCS. In: Proceedings of the conference on design, automation and test in Europe (DATE'07). EDA Consortium, San Jose, pp 1659–1664
30. Das S, Roberts D, Lee S, Pant S, Blaauw D, Austin T, Flautner K, Mudge T (2006) A self-tuning dvs processor using delay-error detection and correction. IEEE J Solid State Circuits 41(4):792–804. doi:10.1109/JSSC.2006.870912
31. Djahromi A, Eltawil A, Kurdahi F, Kanj R (2007) Cross layer error exploitation for aggressive voltage scaling. In: 8th international symposium on quality electronic design, ISQED '07, pp 192–197. doi:10.1109/ISQED.2007.53
32. Donald J, Martonosi M (2005) Leveraging simultaneous multithreading for adaptive thermal control. In: Proceedings of the second workshop on temperature-aware computer systems
33. Donald J, Martonosi M (2006) Techniques for multicore thermal management: classification and new exploration. ACM SIGARCH computer architecture news. 34(2). IEEE computer society
34. Eltawil A, Grayver E, Zou H, Frigon J, Poberezhskiy G, Daneshrad B (2003) Dual antenna UMTS mobile station transceiver asic for 2 mb/s data rate. In: IEEE international,

solid-state circuits conference on Digest of technical papers, ISSCC 2003, vol 1, pp 146–484. doi:10.1109/ISSCC.2003.1234242

35. Ernst D, Das S, Lee S, Blaauw D, Austin T, Mudge T, Kim NS, Flautner K (2004) Razor: circuit-level correction of timing errors for low-power operation. IEEE Micro 24(6):10–20. doi:10.1109/MM.2004.85
36. Gasteier M, Glesner M (1996) Bus-based communication synthesis on system-level. In: Proceedings of 9th international symposium on system synthesis, pp 65–70. doi:10.1109/ISSS.1996.565880
37. Gerards M, Hurink JL, Kuper J (2015) On the interplay between global DVFS and scheduling tasks with precedence constraints. IEEE Trans Comput 64(6):1742–1754
38. Gronowski P, Bowhill W, Preston R, Gowan M, Allmon R (1998) High-performance microprocessor design. IEEE J Solid State Circuits 33:676–686
39. Gunther S, Binns F, Carmean D, Hall J (2001) Managing the impact of increasing microprocessor power consumption. Intel Technol J 5:1–9
40. Herbert S, Marculescu D (2007) Analysis of dynamic voltage/frequency scaling in chip multiprocessors. In: ACM/IEEE international symposium on low power electronics and design (ISLPED). IEEE
41. Hoffmann H, Sidiroglou S, Carbin M, Misailovic S, Agarwal A, Rinard M (2011) Dynamic knobs for responsive power-aware computing. In: Proceedings of the sixteenth international conference on architectural support for programming languages and operating systems, ASPLOS XVI. ACM, New York, pp 199–212. doi:10.1145/1950365.1950390
42. HP Labs (2015) CACTI – An integrated cache and memory access time, cycle time, area, leakage, and dynamic power model. http://www.hpl.hp.com/research/cacti/
43. Huang W et al (2006) Hotspot: a compact thermal modeling methodology for early-stage VLSI design. IEEE Trans Very Large Scale Integr (VLSI) Syst 14(5):501–513. doi:10.1109/TVLSI.2006.876103
44. Hussien A, Khairy M, Khajeh A, Amiri K, Eltawil A, Kurdahi F (2010) A combined channel and hardware noise resilient Viterbi decoder. In: 2010 conference record of the forty fourth Asilomar conference on signals, systems and computers (ASILOMAR), pp 395–399. doi:10.1109/ACSSC.2010.5757543
45. Im S, Banerjee K (2000) Full chip thermal analysis of planar (2-D) and vertically integrated (3-D) high performance ICs. In: International electron devices meeting 2000. Technical digest. IEDM (Cat. No.00CH37138), San Francisco, pp 727–730.
46. ITRS International roadmap of semiconductors. http://www.itrs.net/
47. John JK, Hu JS, Ziavras SG (2005) Optimizing the thermal behavior of subarrayed data caches. In: Proceedings of the 2005 international conference on computer design (ICCD'05). IEEE computer society, Washington, pp 625–630. https://doi.org/10.1109/ICCD.2005.81
48. Kavvadias N, Neofotistos P, Nikolaidis S, Kosmatopoulos K, Laopoulos T (2003) Measurements analysis of the software-related power consumption in microprocessors. In: Proceedings of the 20th IEEE instrumentation and measurement technology conference, IMTC '03, vol 2, pp 981–986. doi:10.1109/IMTC.2003.1207899
49. Kaxiras S, Ju Z, Martonosi M (2001) Cache decay: exploiting generational behavior to reduce cache leakage power. In: Proceedings of the 28th annual international symposium on computer architecture (ISCA'01). ACM, New York, pp 240–251. http://dx.doi.org/10.1145/379240.379268
50. Khajeh A, Cheng SY, Eltawil A, Kurdahi F (2007) Power management for cognitive radio platforms. In: Global telecommunications conference, GLOBECOM '07. IEEE, pp 4066–4070. doi:10.1109/GLOCOM.2007.773
51. Khajeh A, Eltawil A, Kurdahi F (2011) Embedded memories fault-tolerant pre- and post-silicon optimization. IEEE Trans Very Large Scale Integr (VLSI) Syst 19(10):1916–1921. doi:10.1109/TVLSI.2010.2056397
52. Khajeh A, Kim M, Dutt N, Eltawil AM, Kurdahi FJ (2012) Error-aware algorithm/architecture coexploration for video over wireless applications. ACM Trans Embed Comput Syst 11S(1):15:1–15:23. doi:10.1145/2180887.2180892

53. Kumar A, Shang L, Peh L, Jha NK (2008) System-level dynamic thermal management for high-performance microprocessors. IEEE Trans Comput-Aided Des Integr Circuits Syst 27(1):96–108
54. Kurdahi F, Eltawil A, Yi K, Cheng S, Khajeh A (2010) Low-power multimedia system design by aggressive voltage scaling. IEEE Trans Very Large Scale Integr (VLSI) Syst 18(5): 852–856. doi:10.1109/TVLSI.2009.2016665
55. Lahiri K, Raghunathan A, Lakshminarayana G, Dey S (2004) Design of high-performance system-on-chips using communication architecture tuners. IEEE Trans Comput Aided Des Integr Circuits Syst 23(5):620–636. doi:10.1109/TCAD.2004.826585
56. Lee KJ, Skadron K (2005) Using performance counters for runtime temperature sensing in high-performance processors. In: 19th IEEE international parallel and distributed processing symposium, pp 8. doi:10.1109/IPDPS.2005.448
57. Lee S, Das S, Pham T, Austin T, Blaauw D, Mudge T (2004) Reducing pipeline energy demands with local DVS and dynamic retiming. In: Proceedings of the 2004 international symposium on low power electronics and design, ISLPED '04, Newport Beach, pp 319–324.
58. Lee I, Kim H, Yang P, Yoo S, Chung EY, Choi KM, Kong JT, Eo SK (2006) Powervip: SoC power estimation framework at transaction level. In: Asia and South Pacific conference on design automation, pp 8. doi:10.1109/ASPDAC.2006.1594743
59. Leem L, Cho H, Bau J, Jacobson Q, Mitra S (2010) Ersa: error resilient system architecture for probabilistic applications. In: Design, automation test in Europe conference exhibition (DATE), pp 1560–1565. doi:10.1109/DATE.2010.5457059
60. Lienig J (2013) Electromigration and its impact on physical design in future technologies. In: Proceedings of the 2013 ACM international symposium on physical design. ACM, 2013
61. Liou JJ, Krstic A, Jiang YM, Cheng KT (2000) Path selection and pattern generation for dynamic timing analysis considering power supply noise effects. In: IEEE/ACM international conference on computer aided design, ICCAD-2000, pp 493–496. doi:10.1109/ICCAD.2000.896521
62. Lloyd JR (1991) Electromigration failure. J Appl Phys 69:7601–7604
63. Long J, Memik S, Memik G, Mukherjee R (2008) Thermal monitoring mechanisms for chip multiprocessors. ACM Trans Archit Code Optim (TACO) 5(2):9
64. Macii E, Pedram M, Somenzi F (1998) High-level power modeling, estimation, and optimization. IEEE Trans Comput Aided Des Integr Circuits Syst 17(11):1061–1079. doi:10.1109/43.736181
65. MacQueen J (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol 1, no 14
66. Makhzan M, Khajeh A, Eltawil A, Kurdahi F (2007) Limits on voltage scaling for caches utilizing fault tolerant techniques. In: 25th international conference on computer design, ICCD 2007, pp 488–495. doi:10.1109/ICCD.2007.4601943
67. Mamidipaka M, Khouri K, Dutt N, Abadir M (2004) Analytical models for leakage power estimation of memory array structures. In: International conference on hardware/software codesign and system synthesis, CODES + ISSS 2004, pp 146–151. doi:10.1109/CODESS.2004.240909
68. Markovic D, Stojanovic V, Nikolic B, Horowitz M, Brodersen R (2004) Methods for true energy-performance optimization. IEEE J Solid-State Circuits 39(8):1282–1293. doi:10.1109/JSSC.2004.831796
69. Memik SO, Mukherjee R, Ni M, Long J (2008) Optimizing thermal sensor allocation for microprocessors. IEEE Trans Comput Aided Des Integr Circuits Syst 27: 516–527
70. Misailovic S, Roy D, Rinard M (2011) Probabilistically accurate program transformations. In: Yahav E (ed) Static Analysis. Lecture notes in computer science, vol 6887. Springer, Berlin/Heidelberg, pp 316–333. doi:10.1007/978-3-642-23702-7_24
71. Mukherjee R, Memik SO (2006) Systematic temperature sensor allocation and placement for microprocessors. In: Proceedings of the 43rd annual design automation conference. ACM

72. Mukherjee R, Mondal S, Memik S (2006) Thermal sensor allocation and placement for reconfigurable systems. In: IEEE/ACM international conference on computer-aided design (ICCAD'06). IEEE
73. Mukhopadhyay S, Mahmoodi H, Roy K (2004) Statistical design and optimization of SRAM cell for yield enhancement. In: IEEE/ACM international conference on computer aided design, ICCAD-2004, pp 10–13. doi:10.1109/ICCAD.2004.1382534
74. Mukhopadhyay S, Kim K, Mahmoodi H, Roy K (2007) Design of a process variation tolerant self-repairing SRAM for yield enhancement in nanoscaled CMOS. IEEE J Solid-State Circuits 42(6):1370–1382. doi:10.1109/JSSC.2007.897161
75. Mukhopadhyay S, Mahmoodi H, Roy K (2008) Reduction of parametric failures in sub-100-nm SRAM array using body bias. IEEE Trans Comput Aided Des Integr Circuits Syst 27(1): 174–183. doi:10.1109/TCAD.2007.906995
76. Noble B (2000) System support for mobile, adaptive applications. IEEE Pers Commun 7(1):44–49. doi:10.1109/98.824577
77. Nowroz A, Cochran R, Reda S (2010) Thermal monitoring of real processors: techniques for sensor allocation and full characterization. In: Proceedings of the 47th design automation conference. ACM
78. Onouchi M, Yamada T, Morikawa K, Mochizuki I, Sekine H (2006) A system-level power-estimation methodology based on ip-level modeling, power-level adjustment, and power accumulation. In: Asia and South Pacific conference on design automation, pp 4. doi:10.1109/ASPDAC.2006.1594742
79. Orio Rd, Ceric H, Selberherr S (2010) Physically based models of electromigration: from black's equation to modern TCAD models. Microelectron Reliab 50:775–789
80. Park YH, Pasricha S, Kurdahi F, Dutt N (2007) System level power estimation methodology with h.264 decoder prediction IP case study. In: 25th international conference on computer design, ICCD 2007, pp 601–608. doi:10.1109/ICCD.2007.4601959
81. Park Y, Pasricha S, Kurdahi F, Dutt N (2008) Methodology for multi-granularity embedded processor power model generation for an ESL design flow. IEEE/ACM CODES+ISSS
82. Pasricha S, Dutt N, Ben-Romdhane M (2004) Extending the transaction level modeling approach for fast communication architecture exploration. In: Proceedings of 41st design automation conference, New York, pp 113–118
83. Pasricha S, Dutt N, Ben-Romdhane M (2006) Constraint-driven bus matrix synthesis for MPSoC. In: Asia and South Pacific conference on design automation, pp 6. doi:10.1109/ASP-DAC.2006.1594641
84. Pasricha S, Park YH, Kurdahi F, Dutt N (2006) System-level power-performance trade-offs in bus matrix communication architecture synthesis. In: Proceedings of the 4th international conference hardware/Software codesign and system synthesis, CODES + ISSS '06, pp 300–305. doi:10.1145/1176254.1176327
85. Pinto A, Carloni L, Sangiovanni-Vincentelli A (2003) Efficient synthesis of networks on chip. In: Proceedings of 21st international conference on computer design, pp 146–150. doi:10.1109/ICCD.2003.1240887
86. Powell MD, Biswas A, Emer JS, Mukherjee S, Sheikh B, Yardi S (2009) Camp: a technique to estimate per-structure power at run-time using a few simple parameters. In: IEEE 15th international symposium on high performance computer architecture (HPCA'09). IEEE
87. Rabaey JM (1996) Digital integrated circuits: a design perspective. Prentice-Hall, Inc., Upper Saddle River
88. Rao R, Vrudhula S, Chakrabarti C (2007) Throughput of multi-core processors under thermal constraints. In: Proceedings of the 2007 international symposium on low power electronics and design. ACM
89. Ravi S, Raghunathan A, Chakradhar S (2003) Efficient RTL power estimation for large designs. In: Proceedings of 16th international conference on VLSI design, pp 431–439. doi:10.1109/ICVD.2003.1183173
90. Rinard M (2006) Probabilistic accuracy bounds for fault-tolerant computations that discard tasks. In: Proceedings of the 20th annual international conference on supercomputing, ICS '06. ACM, New York, pp 324–334. doi:10.1145/1183401.1183447.

91. Rinard M, Hoffmann H, Misailovic S, Sidiroglou S (2010) Patterns and statistical analysis for understanding reduced resource computing. In: Proceedings of the ACM international conference on object oriented programming systems languages and applications, OOPSLA '10. ACM, New York, pp 806–821. doi:10.1145/1869459.1869525
92. Rodriguez S, Jacob B (2006) Energy/power breakdown of pipelined nanometer caches (90 nm/65 nm/45 nm/32 nm). In: Proceedings of the 2006 international symposium on low power electronics and design, ISLPED'06, pp 25–30. doi:10.1109/LPE.2006.4271802
93. Rohou E, Smith M (1999) Dynamically managing processor temperature and power. In: 2nd workshop on feedback-directed optimization
94. Sami M, Sciuto D, Silvano C, Zaccaria V (2000) Instruction-level power estimation for embedded VLIW cores. In: Proceedings of the eighth international workshop on hardware/-software codesign, CODES 2000, San Diego, pp 34–38
95. Sampson A, Dietl W, Fortuna E, Gnanapragasam D, Ceze L, Grossman D (2011) Enerj: approximate data types for safe and general low-power computation. In: Proceedings of the 32nd ACM SIGPLAN conference on programming language design and implementation, PLDI '11. ACM, New York, pp 164–174. doi:10.1145/1993498.1993518
96. Sanchez H, Philip R, Alvarez J, Gerosa G (1997) A CMOS temperature sensor for PowerPC RISC microprocessors. In: Proceedings of the symposium on VLSI circuits. IEEE, pp 13–14
97. Sarrigeorgidis K, Rabaey J (2004) Ultra low power cordic processor for wireless communication algorithms. J VLSI Signal Process Syst Signal Image Video Technol 38(2):115–130. doi:10.1023/B:VLSI.0000040424.11334.34
98. Sarta D, Trifone D, Ascia G (1999) A data dependent approach to instruction level power estimation. In: Proceedings of IEEE Alessandro volta memorial workshop on low-power design, pp 182–190. doi:10.1109/LPD.1999.750419
99. Shanbhag N(2002) Reliable and energy-efficient digital signal processing. In: Proceedings of 39th design automation conference, pp 830–835. doi:10.1109/DAC.2002.1012737
100. Shatzkes M, Lloyd JR (1986) A model for conductor failure considering diffusion concurrently with electromigration resulting in a current exponent of 2. J Appl Phys 59, 3890–3893
101. Shin JY, Kurdahi F, Dutt N (2015) Cooperative on-chip temperature estimationusing multiple virtual sensors. IEEE Embed Syst Lett 7(2):37–40. doi:10.1109/LES.2015.2400992
102. Skadron K, Stan MR, Huang W, Velusamy S, Sankaranarayanan K, Tarjan D (2003) Temperature-aware computer systems: opportunities and challenges. IEEE Micro 23(6): 52–61
103. Skadron K, Stan M, Sankaranarayanan K, Huang W, Velusamy S, Tarjan D (2004) Temperature-aware microarchitecture: modeling and implementation. ACM Trans Archit Code Optim 1:94–125
104. Sloan J, Sartori J, Kumar R (2012) On software design for stochastic processors. In: Proceedings of the 49th annual design automation conference, DAC '12. ACM, New York, pp 918–923. doi:10.1145/2228360.2228524
105. Uht A (2004) Going beyond worst-case specs with teatime. Computer 37(3):51–56. doi:10.1109/MC.2004.1274004
106. Wan L, Chen D (2010) Analysis of circuit dynamic behavior with timed ternary decision diagram. In: 2010 IEEE/ACM international conference on computer-aided design (ICCAD), pp 516–523. doi:10.1109/ICCAD.2010.5653852
107. Wang H, Tan S, Swarup S, Liu X (2013) A power-driven thermal sensor placement algorithm for dynamic thermal management. In: Design, automation & test in Europe conference & exhibition (DATE'13). IEEE
108. Wu W, Jin L, Yang J, Liu P, Tan S (2006) A systematic method for functional unit power estimation in mircoprocessors. In: 2006 43rd ACM/IEEE on design automation conference. IEEE
109. Ye W, Vijaykrishnan N, Kandemir M, Irwin M (2000) The design and use of simplepower: a cycle-accurate energy estimation tool. In: Proceedings of 2000 design automation conference, pp 340–345. doi:10.1109/DAC.2000.855333

110. Zaynoun S, Khairy M, Eltawil A, Kurdahi F, Khajeh A (2012) Fast error aware model for arithmetic and logic circuits. In: 2012 IEEE 30th international conference on computer design (ICCD), pp 322–328. doi:10.1109/ICCD.2012.6378659

111. Zhang Y, Li Y, Li X, Yao SC (2013) Strip-and-zone micro-channel liquid cooling of integrated circuits chips with non-uniform power distributions. In: ASME 2013 heat transfer summer conference

112. Zhang Y, Shi B, Srivastava A (2010) A statistical framework for designing on-chip thermal sensing infrastructure in nano-scale systems. IEEE Trans Very Large Scale Integration (VLSI) Syst 22(2):270–279