CHAPTER

5 Models for Power and Thermal Estimation

In the previous chapter, we examined models for the early performance estimation of on-chip communication architectures. In addition to performance, power is another important architectural design constraint, that is beginning to dominate multiprocessor SoC (MPSoC) design, particularly for mobile applications such as cellular phones, MP3 players, and laptops. These portable devices run on batteries that have a limited energy budget between charges. MPSoC designs intended for use in such portable scenarios must have lower power consumption, to improve user experience. Reducing power consumption is also a priority for non-portable MPSoC applications, such as those used in server farms that tend to consume significant amounts of power (e.g., as much as 2 megawatts for a 25,000 square foot server farm with 8000 servers [1]). According to SIA [79] projections of future silicon technologies, the operating frequency and transistor density of MPSoCs will continue to increase, making power dissipation for these highly integrated and complex designs a major concern.

Excessive power dissipation has an undesirable thermal side effect of increasing device temperature that can adversely affect the *reliability* of MPSoCs. Reliability is the probability that the system will operate correctly at any given time of operation, and is measured by *mean time to failure* (MTTF). It is an important design criterion to extend MTTF beyond the expected useful life of a product, especially for critical systems such as those used in aircraft control, automotive control, medical equipment, and defense applications. Temperature cycles and spikes due to excessive power dissipation can induce mechanical stress and dielectric breakdown that can cause device failure. Commonly used techniques to reduce power dissipation such as voltage scaling, although beneficial, can also lead to issues with signal integrity (making it harder to guarantee error-free data transfers). As technology scales into the deep submicron (DSM) region, reduced voltage supply levels, increased leakage power, external electromagnetic interference (EMI), crosstalk, and soft errors will further reduce signal integrity. Voltage scaling will thus not be sufficient to mitigate the power dissipation problem. Expensive cooling and packaging equipment will be required to keep MPSoCs functioning at a reasonable temperature. Techniques to estimate and reduce power consumption will therefore become essential for lowering operating temperatures, so that the MTTF is increased and cooling and packaging costs are reduced.

Thus, with scaling trends in emerging technologies, and the increasing proliferation of the Internet and mobile computing, the power problem has assumed a critical status that cannot be ignored by MPSoC designers. On-chip communication architectures have a considerable impact on MPSoC power consumption. There are several reasons why the interconnect fabric is receiving so much attention with respect to power consumption [1, 2]. First, unlike transistors, interconnects have not scaled exponentially in DSM technologies, as a result of which interconnect capacitance forms a larger portion of total chip capacitance [3]. Second, the problem of modeling DSM effects could be largely ignored in pre-DSM technologies, where transistors were the main focus due to their large sizes. However, in DSM technologies, effects such as coupling capacitance between adjacent wires become increasingly dominant [4, 5]. Third, interconnects in today's designs are proportionally longer, which implies that interconnect delay has increased. Fourth, the use of a large number of repeaters and vias to reduce wire delay almost doubles power consumption in interconnects [6]. Finally, state of the art communication architectures consist not only of bus wires, but also significant amounts of hardware logic (e.g., bridges, arbiters, decoders, buffers, etc.) that is comparable to the amount of logic in embedded processors of moderate complexity [7]. It has been predicted that communication architectures will consume a larger portion of on-chip power in future technologies [8]. There is, therefore, a need to create models for estimating power consumption of on-chip communication architectures as early as possible in a design flow, to better design and optimize MPSoCs. Additionally, the thermal effects of power dissipation cannot be ignored, since they are beginning to have a significant impact on the power, performance, design, and reliability of on-chip buses [9].

In this chapter, we present models for on-chip communication architecture power and thermal estimation. To address the problem of excessive power dissipation, designers need such models to evaluate the impact of design decisions on chip power dissipation. Figure 5.1 shows the positioning of communication architecture power and thermal estimation models in a typical electronic system level (ESL) design flow. These models inevitably require extrapolating information from lower levels in the design flow up to the system level. Such estimation models allow designers to make optimizations to possibly reduce chip power dissipation early in the design flow, where design decisions have a greater impact. In Section 5.1, we present models for power estimation of bus wires, including DSM-aware bus wire power models that take into account crosstalk and coupling capacitance between adjacent wires. Section 5.2 elaborates on approaches that attempt to estimate the power consumption of the entire bus-based on-chip communication architecture, including that of bus logic components. Section 5.3 presents models for thermal evaluation of bus wires. Finally, Section 5.4 presents a discussion of PVT (process, voltage, temperature) variation-aware power estimation for on-chip communication architectures in ultra DSM (UDSM) technologies.

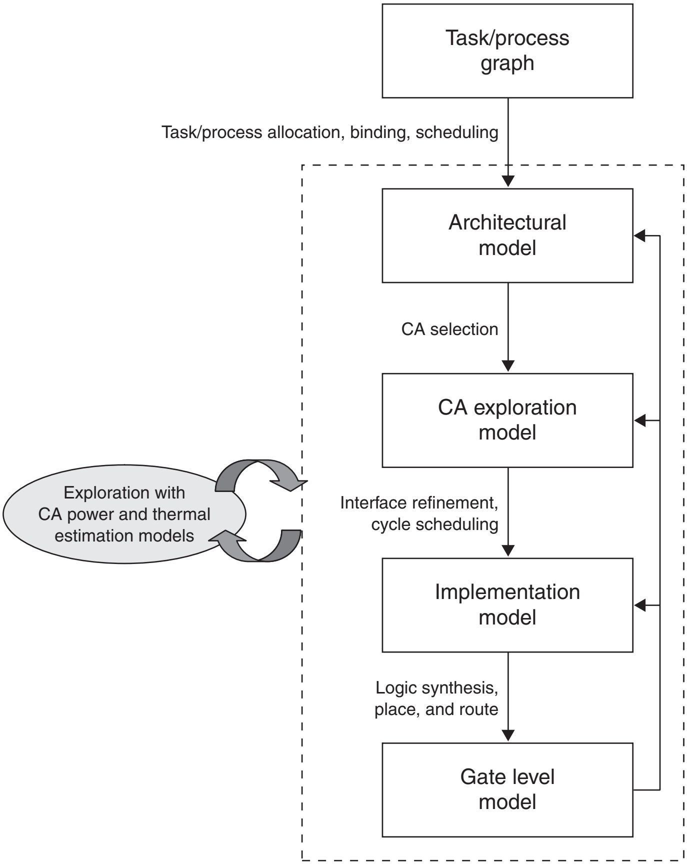

Task/process graph
Task/process allocation, binding, scheduling
Architectural model
CA selection
CA exploration model
Exploration with CA power and thermal estimation models
Interface refinement, cycle scheduling
Implementation model
Logic synthesis, place, and route
Gate level model

FIGURE 5.1

Communication architecture (CA) power and thermal estimation in a typical ESL design flow

5.1 BUS WIRE POWER MODELS

Complementary metal-oxide semiconductor (CMOS) is the dominant technology for designing system-on-chips (SoCs). Power consumption in CMOS logic circuits can be expressed by the following general equation [1]:

$$P = ACV^2 f + \tau AVI_{\text{short}} f + VI_{\text{leak}} \tag{5.1}$$

The equation has three components. The first term in the equation represents the *dynamic* power consumption due to the charging and discharging of the capacitive load on each logic gate's output. The dynamic power consumption is proportional to the frequency of the system's operation *f*, the activity of the gates of the system *A*, the square of the supply voltage *V*, and the total capacitance seen by the gate's output *C*. The second term in the equation represents power consumption due to *short circuit* current I_{short} that flows for a short instant of time T between the supply voltage and the ground when the output of a CMOS logic gate switches. The third term represents power consumption due to *leakage* current I_{leak} irrespective of the state of the system. In pre-DSM technologies, the first term dominated overall power consumption. However, as CMOS process technology shrinks toward DSM, the leakage term has started to play a more prominent role.

Like logic gates, wires have a capacitance associated with them, representing charge that must be added or removed to change the electrical potential on a wire. Whenever a data bit is transmitted on a bus wire, the charging and discharging of this wire capacitance results in power consumption. An accurate modeling of wire capacitance is, however, a non-trivial task and still an ongoing subject of advanced research [3]. Complications in estimation arise because the structure of a wire in contemporary integrated circuits (ICs) is 3-D, making the capacitance of such a wire a function of its shape, environment, its distance to the substrate, and its distance to the surrounding wires. Typically, designers use simple first order models to provide a basic understanding of the nature of the wire capacitance and its parameters. If a completed layout is available, designers use advanced extraction tools to obtain values of wire capacitance. Semiconductor manufacturers also often provide empirical data for various contributors to wire capacitance, as measured for several test dies.

5.1.1 Early Work

A lot of early work on high level bus power estimation [10–16] and bus encoding to reduce communication power consumption [17–21] (covered in more detail in Chapter 7) considered a simple model of bus power consumption that, as Eq. (5.2) shows, primarily focused on the contribution of dynamic switching power on a wire. The bus power consumption model can be expressed as:

$$P_{\text{AVG}} = \frac{1}{2} C_{\text{bus}} V_{\text{dd}}^2 n_{\text{trans}} f \tag{5.2}$$

where C_{bus} is the total bus capacitance (cumulative capacitance of all the wires in the bus), V_{dd} is the power supply voltage, n_{trans} is the average number of transitions on the bus, and f is the operating frequency of the system bus. Let B^t be the value of the bit string on the bus at time t, and let L be the total length of the data stream transmitted on the bus. Then the average number of transitions on the bus can be expressed as:

$$n_{\text{trans}} = \frac{\sum_{t=0}^{L-1} HD(B^t, B^{t+1})}{L-1} \tag{5.3}$$

where $HD(B^t, B^{t+1})$ is the Hamming distance between the words on the system bus times at times t and $t+1$. For example, for a 16 bit wide bus, if the word on the bus at time t is 1000100011001100, and the word on the bus at time $t+1$ is 1001100110001000, then the Hamming distance is the number of bit-flips between the two words, which is four.

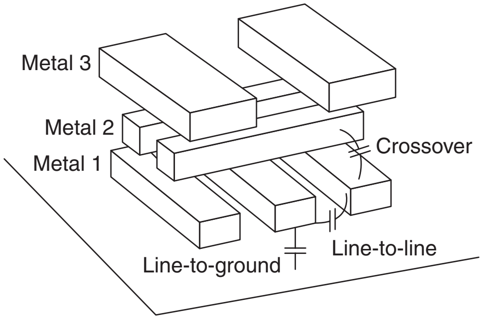


FIGURE 5.2

Three primary wire capacitance components: line-to-line, line-to-ground, and crossover capacitances [22]
© 1992 IEEE

An early model for wire capacitance was proposed by Chern et al. [22]. Figure 5.2 shows the three primary constituents of wire capacitance – line-to-line capacitance, line-to-ground capacitance, and crossover capacitance. The crossover capacitance is the capacitance between wires in different metal layers (since typical SoC designs are fabricated on multiple metal layers). The simplest of three constituents, the line-to-ground capacitance, was calculated as [22]:

$$\frac{C}{\epsilon} = \frac{W}{H} + 3.28\left(\frac{T}{T+2H}\right)^{0.023} + \left(\frac{S}{S+2H}\right)^{1.16} \tag{5.4}$$

where W is the metal width, S is the space between two lines (or wires), T is the thickness of the metal, H is the thickness of dielectric layer between metal layers, and ε is the dielectric constant. Similar expressions were used for the other two constituents of wire capacitance. The total capacitance of a wire is the sum of all these constituents. As an example, consider a wire in the metal 2 layer, in Fig. 5.2. The total capacitance of the metal 2 line C_2 is given by:

$$C_2 = C_{21} + C_{23} + C_{22} \tag{5.5}$$

where C_{21} is the capacitance from the metal 2 later to the metal 1 layer, C_{23} is the capacitance from the metal 2 later to the metal 3 layer, and C_{22} is the capacitance between wires in the metal 2 layer. C_{21} and C_{23} are crossover capacitances, and C_{22} is a line-to-line capacitance. Fairly complex, but comprehensive expressions that are functions of W, S, T, H, and ε (whose values can be obtained from technology library data sheets), for both of these constituent capacitances are presented in the appendix of [22]. Comparisons of the accuracy of these models with accurate numerical simulations and measurements by Sakurai et al. [23] indicated an accuracy of within 8% for capacitance values.

There are many analytical models that approximate the capacitance of a wire over a plane. More accurate models combine a bottom plate term with a fringing term to account for field lines originating from the edge and top of the wire [24]. Since wires today are becoming taller rather than wider, in order to reduce resistance

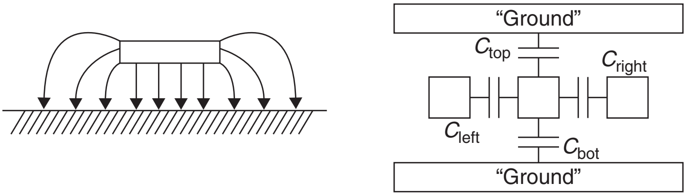


FIGURE 5.3

Isolated and realistic wire capacitance models [24]
© 2001 IEEE

with technology scaling, their side-to-side capacitances are growing and becoming more significant. As such, capacitance is better modeled by four parallel plate capacitors for the top, bottom, left, and right sides, plus a constant term for fringing capacitance, as shown in Fig. 5.3 [24, 25]. The vertical and horizontal capacitors can have different relative dielectrics for technologies that use *low K* materials, and the wire capacitance (C_{wire}) can be expressed as [24, 26]:

$$C_{\text{wire}} = \epsilon_0 \left(2K\epsilon_{\text{horiz}} \frac{\text{thick}}{\text{spacing}} + 2\epsilon_{\text{vert}} \frac{\text{width}}{ILD_{\text{thick}}} \right) + \text{fringe}(\epsilon_{\text{horiz}}, \epsilon_{\text{vert}}) \tag{5.6}$$

where ε_0 is the electric constant, $\varepsilon_{\text{horiz}}$ and $\varepsilon_{\text{vert}}$ are the horizontal and vertical relative dielectric constants, respectively, and ILD_{thick} is the thickness of the interlayer dielectric (ILD). The plates at the top and bottom are typically modeled as grounded, since they represent orthogonally routed conductors, that averaged over the length of a wire, maintain a constant voltage (the capacitance would be multiplied by an appropriate factor if the orthogonal wires switched simultaneously and monotonically). Capacitances to the left and right on the other hand have data dependent effective capacitances that can vary. If the left and right neighbors of a wire switch in the opposite direction as the wire, then the effective side capacitances double, otherwise if they switch with the wire, the effective side capacitance approaches zero. This effect is termed as *Miller multiplication* and is modeled by varying the *K* parameter between 0 and 2 in Eq. (5.6). The *fringe* term depends weakly on geometry. Simplifications to the model can be made for top metal wires, for which only three parallel plates and fringing terms on the horizontal capacitors need to be considered.

5.1.2 Coupling-Aware Power Models

A significant amount of work has been done to model the capacitive (and inductive) parasitic interactions and coupling of bus wires in DSM technologies [2, 4, 5, 27–40]. The effects of coupling between wires not only tend to dominate line-to-ground coupling, but also introduce dependencies in the energy drawn from the power supply by their drivers. A compact energy model, based on a distributed circuit model for DSM buses was presented by Sotiriadis et al. [5]. Figure 5.4 shows a DSM bus that consists of several parallel lines driven by CMOS inverters and with repeaters inserted to reduce signal propagation delay. Figure 5.5 shows an

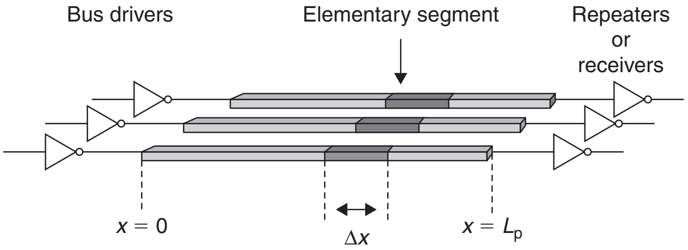


FIGURE 5.4

DSM bus [5]
© 2002 IEEE

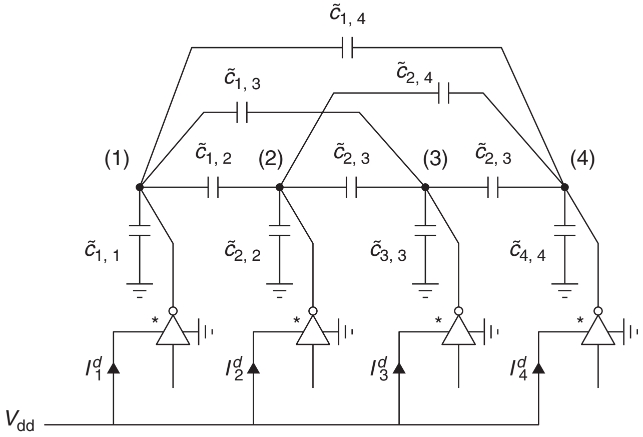


FIGURE 5.5

Lumped energy equivalent DSM bus model [5]
© 2002 IEEE

equivalent capacitive network model for the DSM bus used in [5], called the ***lumped energy equivalent DSM*** bus model, that represents all the capacitances for a set of wires (the figure shows a model for $n=4$ wires). The energy drawn during a transition, from the power supply by the bus drivers, for a bus with n lines is given by the expression:

$$E = \sum_{i=1}^{n} V_i^f e_i^T C^t (V^f - V^i) \tag{5.7}$$

where $V^i = [V_1^i, V_2^i, \ldots, V_n^i]$ and $V^f = [V_1^f, V_2^f, \ldots, V_n^f]$ are vectors with n coordinates representing the initial and final voltages on the bus lines in the course of the transition, e_{i} is a vector with a 1 in the ith position, and 0 elsewhere, and C^t is the total capacitance conductance matrix (which represents the capacitances between the lines and ground). Since, traditionally, bus lines are laid parallel and co-planar, most of the electric field is trapped between the adjacent lines

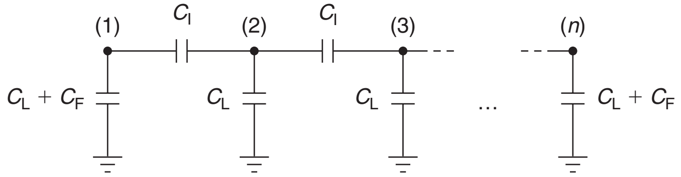


FIGURE 5.6

Simple approximate energy model for DSM bus [5]
© 2002 IEEE

and ground. The capacitance between non-adjacent lines is, thus, practically negligible compared to capacitance between adjacent lines, or the capacitance between the line and the ground. An approximate bus energy model can ignore parasitics between non-adjacent lines [3, 27, 30, 33]. Additionally, assuming that all the grounded capacitors have the same value (except for the boundary ones due to fringing effects) and that all interline capacitances are also the same, the approximate DSM bus model becomes that shown in Fig. 5.6 and the value of the approximate total capacitance conductance matrix C^{ta} (approximate C^t) is given as:

$$C^{ta} = \begin{bmatrix} 1+\lambda+\zeta & -\lambda & 0 & \cdots & 0 \\ -\lambda & 1+2\lambda & -\lambda & \cdots & 0 \\ 0 & -\lambda & 1+2\lambda & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1+\lambda+\zeta \end{bmatrix} C_L \tag{5.8}$$

where

$$\lambda = \frac{C_I}{C_L}, \quad \zeta = \frac{C_F}{C_L} \tag{5.9}$$

C_L is the line capacitance (includes the capacitance of the driver and the receiver), C_F the fringing capacitance, and C_I the inter-line capacitance. The parameters λ and ζ depend on the particular CMOS technology used, as well as the specific geometry, metal layer, and shielding of the bus. Also, the C_F term can be ignored for wide buses since it does not contribute significantly to overall energy consumption.

A parameterizable analytical model for power dissipation on the bus was presented by Kretzschmar et al. [39]. This model was validated by power simulation with layout (including parasitics) of a particular bus implementation. The model of a bus line used is shown in Fig. 5.7. The line capacitance C_{line} of the wire encapsulates the ground and coupling capacitances of the wires (C_{wire}), in addition to the internal (C_{int}) and input capacitances (C_{inp}) of the active elements on the wire such as the drivers, repeaters, and receivers. As discussed earlier, due to DSM, the wire capacitance is determined not only by vertical capacitance C_{vertical} but also increasingly by the coupling capacitances C_{lateral} to adjacent wires. Due to the Miller effect, the power consumption of the wire becomes data dependent – if the neighboring wires switch in the same direction, then the C_{lateral} is not charged; however for opposite switches of neighboring wires, twice C_{lateral} is charged. Assuming an independent distribution of activity on all the bus lines, adjacent lines will switch as often in the same direction as in the opposite direction. On an average, the coupling capacitance between lines is charged once, and therefore it

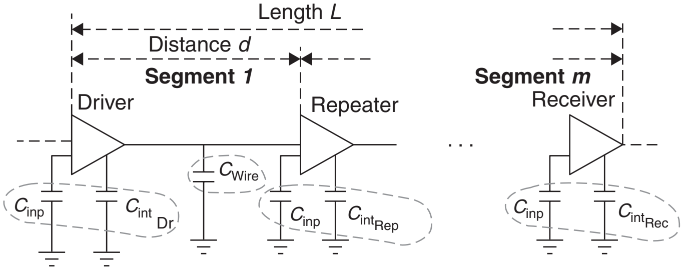


FIGURE 5.7

Model of bus line including driver, repeaters, and receiver [39]
© 2004 IEEE

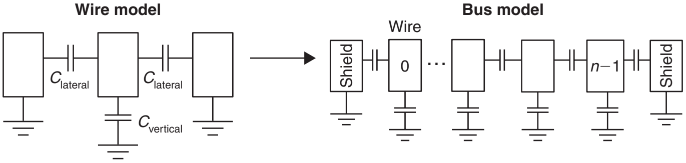


FIGURE 5.8

Bus wire model with two adjacent wires and extension to *n*-wire bus [39]
© 2004 IEEE

is assumed in the model that the two adjacent lines are connected to the ground, as shown in Fig. 5.8. Whenever there is a transition from 0 to 1 on the middle wire, the vertical capacitance and both the adjacent capacitances are charged. The wire capacitance per unit length is thus given by the relation:

$$C_{\text{wire}} = C_{\text{vertical}} + 2C_{\text{lateral}} \tag{5.10}$$

Assuming identical repeaters and drivers, the capacitance of an *n*-wire bus is given by:

$$C_{\text{bus}} = nL(C_{\text{vertical}} + 2C_{\text{lateral}}) + n(\frac{L}{d}C_{\text{int,Dr}} + \frac{L}{d}C_{\text{inp,Dr}} + C_{\text{int,Rec}} + C_{\text{inp,Rec}}) \tag{5.11}$$

where L is the length of the line and d is the inter-repeater distance. This equation can be used to calculate the average capacitance of a bus line per unit length, based on values from a CMOS technology library. The wire capacitance C_{wire} depends on the spacing to adjacent wires as well as the distance to the metal layers above and below. The bus power consumption can be obtained by plugging the capacitance value into Eq. (5.2).

5.1.3 High Level Power Models

A high level power model for the purposes of early design space exploration and incorporation in high level synthesis techniques [41] was proposed by Gupta et al. [2]. The total power consumption of a bus in this model is given by [39]:

$$P_{\text{total}} = P_{\text{sw}} + P_{\text{vias}} + P_{\text{repeaters}} \tag{5.12}$$

where P_{sw} is the power consumption due to switched interconnect capacitance and inter-wire coupling, P_{vias} is the power consumed by the vias due to the use of multiple metal layers, and $P_{\text{repeaters}}$ is the power consumed by repeaters inserted to minimize signal delay. Each of these components is discussed in more detail below.

5.1.3.1 *Switching Power*

The model makes use of a table-lookup method, first presented by Taylor et al. [42], where total switching power is determined by the types of transitions, instead of the number of transitions that can occur on the interconnect. Since coupling effects between wires decrease sharply the further apart they are, only an interconnect and its adjacent wires need to be considered. Table 5.1 shows the set of various transitions that are possible on three-wire interconnects. The general idea is to use low level transistor simulation to construct such a three-wire lookup table for minimally spaced wires of various lengths that gives the power consumption for each type of transition in the transition set. Such a scheme allows an accurate modeling of the electrical characteristics of a wire for a particular CMOS process technology. Also, since only three wires need to be simulated, the required time for low level transistor simulation is negligible. The total interconnect power can then be obtained by counting the types of transitions on the interconnect and performing a table lookup. The model does not, however, consider the effect of glitches and the authors point to techniques presented by Raghunathan et al. [43] as a means to suppress glitches. Also, instead of counting the transitions on the interconnect and performing a table lookup, the authors propose schemes [2] to estimate the types of transitions. This is done because the authors claim that high level design automation tools will consider multiple architectures to implement a design, many of which will be similar except for a few enhancements. As such, it will be useless and time consuming to run a full-fledged simulation on each architecture to characterize the switching activity on its interconnects. Instead, switching activity characterization is performed only once on an architecture that does not change drastically, and then first order estimates of switching activity are used on similar architectures.

5.1.3.2 *Power Due to Vias*

The purpose of vias is to (i) connect transistors residing on the substrate with the interconnect connecting these transistors, and (ii) connect interconnects running on multiple metal layers. The total power consumed by the vias is given by:

$$P_{\text{vias}} = V_{\text{N}} \cdot P_{\text{via}} \tag{5.13}$$

where V_{N} is the number of vias and P_{via} is the power consumption of a single via. The number of vias is estimated using interconnect layout, obtained with a

Table 5.1 Types of transitions on three-wire interconnects [2]

S S S	S X S	S S X	S X O	S X X
X X X	X S X	X X O	O X O	X S O

S = Stationary, X = Transition, O = Opposing transition
© 2003 IEEE

floorplanner and statistical methods, and then counting the number of times an interconnect changes direction. Although the via power P_{via} is dependent on the layer in which it resides, this is not taken into account in the model, which uses approximate values of P_{via} by taking its average for different configurations (or optionally using a weighting factor to each via configuration representing its proportional contribution to all the vias in the layout). The power consumed by vias used in repeaters is not estimated by Eq. (5.13), but is accounted for in the repeater power consumption term, described below.

5.1.3.3 *Power Due to Repeaters*

As mentioned earlier, with shrinking feature size, interconnect wires are getting proportionally longer and not scaling as well as transistors. A wire can be modeled as a simple *RC* network, for which the signal propagation delay is a quadratic function of length, because both resistance and capacitance are functions of length [3]. Inserting repeaters is a commonly used practice to reduce wire delay, since it reduces the quadratic dependence into a linear dependence on wire length [24, 44–46]. The starting and ending coordinates of an interconnect are obtained from a floorplanner, and statistical methods are used to obtain a layout for the interconnect. The number of repeaters (N_{R}) are obtained from formulations presented by Kapur et al. [6] and Bakoglu et al. [45], which give the optimal inter-repeater distance for a given CMOS process technology. The total number of repeater vias (V_{R}) are then calculated to be twice the number of repeaters, since paths are need to descend and ascend from the substrate where the repeaters reside. The total repeater power is then given as:

$$P_{\text{rep}} = C_{\text{rep}} \cdot V_{\text{dd}}^2 \cdot f \cdot \sum_{i \in \text{I}} \rho_i N_{Ri} + P_{\text{via}} \cdot \sum_{i \in \text{I}} V_{Ri} \tag{5.14}$$

where the capacitance of a single repeater C_{rep} is given by the equations in [6], ρ_{i} is the switching activity, V_{dd} is the operating voltage, f is the clock frequency, and P_{via} was described earlier. The authors claim that Eq. (5.14) can be extended to incorporate leakage power in the repeaters.

5.2 COMPREHENSIVE BUS ARCHITECTURE POWER MODELS

In addition to wires, an important component of on-chip communication architectures is the bus logic, which consists of components such as bridges, arbiters, decoders, and buffer stages. A comprehensive, gate level power estimation methodology for the estimation of logic and interconnect power for the AMBA AHB/APB [47] hierarchical on-chip communication architecture was presented by Lahiri et al. [7]. The goal was to analyze the contributions of different components of the communication architecture to the overall communication architecture power consumption. This methodology and the results from the power estimation study are described below.

Figure 5.9 shows the methodology used for communication architecture power estimation. Synopsys CoreTools [48] was used to configure the parameters

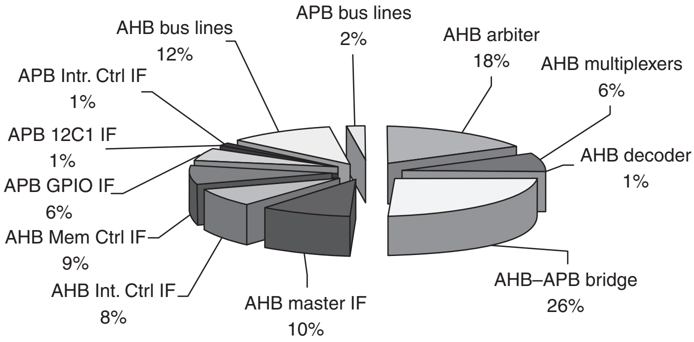


FIGURE 5.10

Breakdown of bus-based communication architecture power consumption for a simple AMBA AHB/APB hierarchical communication architecture [7]
© 2004 IEEE

bus-based system was considered, with two masters and two slaves (a memory controller and an interrupt controller) on the AHB, and three peripherals (general purpose I/O, interrupt controller, and I2C external bus interface) on the APB bus that were connected to the AHB bus via an AHB–APB bridge. The testbench had different transactions, including single transfers and bursts, with the addresses being generated randomly and uniformly. The global wire length was estimated to be between 1.5 and 3 mm, and the overall power consumption of the communication architecture was 12 mW. Figure 5.10 shows a breakdown of the power consumed by the various components of the AMBA bus. It can be seen that the power consumed by the bus lines is only 14% of the overall power, although it is often assumed that bus wires consume a large amount of power. This shows that it is important to consider bus logic when estimating power for an on-chip communication architecture.

5.2.1 Macro-Models for Bus Matrix Communication Architectures

A comprehensive macro-modeling-based power estimation approach for estimating the power consumption of the logic and wires in the AMBA AHB [47] bus matrix communication architecture was proposed by Pasricha et al. [56]. Unlike the approach proposed by Lahiri et al. [7] that estimates power at the gate level, this approach creates reusable power models of the on-chip communication architecture that can be used early in the design flow, at the system level, for fast and accurate power estimation.

The energy consumption of a bus logic (or for that matter any hardware) component can be obtained by identifying factors or events that cause a noticeable change in its energy profile. For this purpose, Pasricha et al. [56] proposed creating energy *macro-models* that can encapsulate factors having a strong correlation to energy consumption for a given component. A macro-model consists of variables that represent factors influencing energy consumption, and regression coefficients that capture the correlation of each of the variables with overall component energy consumption. A general linear energy macro-model for a component can be expressed as:

$$E_{\text{component}} = \alpha_0 + \sum_{i=1}^{n} \alpha_i \cdot \psi_i \tag{5.15}$$

where α_0 is the energy of the component which is independent of the model variables (e.g., leakage, clock energy consumption), and α_i is the regression coefficient for the model variable ψ_i. Note that Eq. (5.15) shows a linear energy macro-model that may not be as accurate as a higher order quadratic model. The authors' motivation for considering a linear model was that if the linear model provided good enough estimation accuracy, there was no need to consider more complex quadratic models (that are typically harder to create and evaluate, making power estimation more time consuming).

Three types of model variables – *control*, *data*, and *structural* – were considered in the energy macro-models. These variables represent different factors influencing energy consumption. The *control* factor represents control events, involving a control signal that triggers energy consumption either when a transition occurs from 1 to 0 or 0 to 1, or when it maintains a value of 0 or 1 for a cycle. Control variables can either have a value of 1 when a control event occurs, or 0 when no event occurs, in the energy macro-model relation Eq. (5.15). The *data* factor represents data events that trigger energy consumption on data value changes. Data variables take an integer value in Eq. (5.15) representing the Hamming distance (number of bit-flips) of successive data inputs. Finally, *structural* factors, such as data bus widths and number of components connected to the input also affect energy consumption of a component. They are represented by their integer values in Eq. (5.15).

A high level overview of the methodology used to create energy macro-models for bus logic components is shown in Fig. 5.11. Initially, a system testbench consisting of masters and slaves interconnected using the AMBA AHB bus matrix [47]

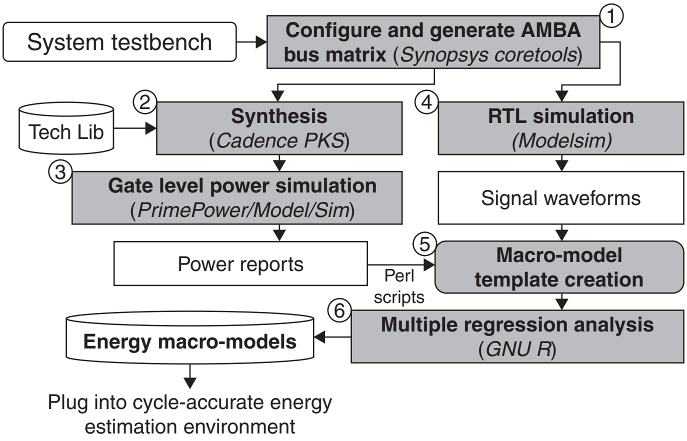

System testbench
1
Configure and generate AMBA bus matrix (Synopsys coretools)
Tech Lib
2
Synthesis (Cadence PKS)
4
RTL simulation (Modelsim)
3
Gate level power simulation (PrimePower/Model/Sim)
Signal waveforms
Power reports
Perl scripts
5
Macro-model template creation
6
Multiple regression analysis (GNU R)
Energy macro-models
Plug into cycle-accurate energy estimation environment

FIGURE 5.11

Energy macro-model generation methodology [56]
© 2006 IEEE

fabric is chosen. The testbench generates traffic patterns consisting of single and burst transactions of varying sizes, and different modes (e.g., SPLIT/RETRY, locked bus, etc.) that exercise the matrix under different operating conditions. Synopsys Coretools [48] is used to configure the bus matrix (i.e., specify data bus width, number of masters and slaves, etc.) and generate a synthesizable RTL description of the bus matrix communication architecture (Step 1). This description is synthesized to the gate level with the Cadence Physically Knowledgeable Synthesis (PKS) [57] tool, for the target standard cell library (Step 2). PKS pre-places cells and derives accurate wire length estimates during logic synthesis. In addition, it generates a clock tree including clock de-skewing buffers. The gate level netlist is then used with Synopsys PrimePower [58] to generate power numbers (Step 3).

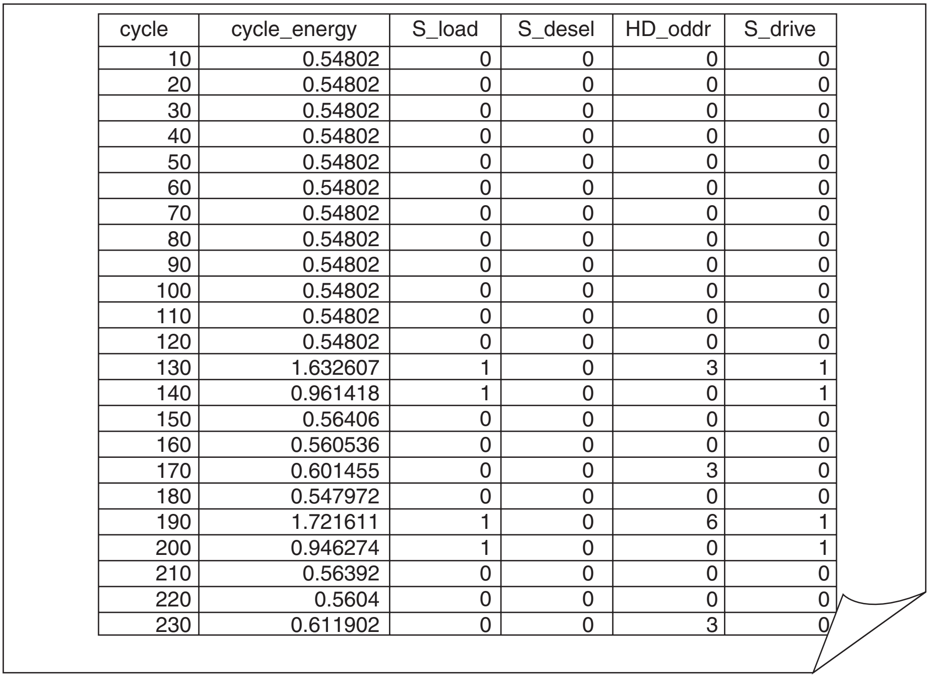
In parallel with the synthesis flow, RTL simulation is performed to generate signal waveform traces for important data and control signals (Step 4). These signal waveforms are compared with cycle energy numbers, obtained after processing PrimePower generated power report files with Perl scripts, to determine which data and control signals in the matrix have a noticeable effect on its energy consumption. Only those signals are considered that change in value when an increase in bus matrix energy consumption of at least 0.01% is observed over the base case (i.e., no data traffic). Note that a finer grained selection criterion (e.g., 0.001%) will result in even more accuracy, but at the cost of more complex macro-models that take longer to create and evaluate. The selected data and control events become the variables in a macro-model template that consists of energy and variable values for each cycle of testbench execution (Step 5). Figure 5.12 shows an example of a macro-model template for one of the components of the bus matrix. The template consists of energy values (*cycle_energy*) and variable values (*S_load, S_desel, HD_addr, S_drive*) for each cycle of testbench execution. This template is used as an input to the GNU R tool [59] that performs multiple linear regression analysis to find coefficient values for the chosen variables (Step 6). Steps 1–6 are repeated for testbenches having different structural attributes such as data bus widths and number of masters and slaves, to identify structural factors (variables) that may influence cycle energy.

Statistical coefficients such as *Multiple-R*, *R-square*, and *standard deviation for residuals* [60] are used to determine the goodness of fit and the strength of the correlation between the cycle energy and the model variables. Once a good fit between cycle energy and macro-model variables is found, the energy macro-models are generated in the final step. These models can then be plugged into any system level cycle-accurate or cycle-approximate simulation environment, to get energy consumption values for the AMBA AHB bus matrix communication architecture.

To obtain the energy consumption for the entire AMBA AHB bus matrix communication architecture, the energy macro-model generation methodology was used to create macro-models for each of its components. The total energy consumption of a bus matrix can be expressed as:

$$E_{\text{MATRIX}} = E_{\text{INP}} + E_{\text{DEC}} + E_{\text{ARB}} + E_{\text{OUT}} + E_{\text{WIRE}} \tag{5.16}$$

where E_{INP} and E_{DEC} are the energy for the input and decoder components for all the masters connected to the matrix, E_{ARB} and E_{OUT} are the energy for arbiters and output stages connecting slaves to the matrix, and E_{WIRE} is the energy of all



cycle	cycle_energy	S_load	S_desel	HD_oddr	S_drive
10	0.54802	0	0	0	0
20	0.54802	0	0	0	0
30	0.54802	0	0	0	0
40	0.54802	0	0	0	0
50	0.54802	0	0	0	0
60	0.54802	0	0	0	0
70	0.54802	0	0	0	0
80	0.54802	0	0	0	0
90	0.54802	0	0	0	0
100	0.54802	0	0	0	0
110	0.54802	0	0	0	0
120	0.54802	0	0	0	0
130	1.632607	1	0	3	1
140	0.961418	1	0	0	1
150	0.56406	0	0	0	0
160	0.560536	0	0	0	0
170	0.601455	0	0	3	0
180	0.547972	0	0	0	0
190	1.721611	1	0	6	1
200	0.946274	1	0	0	1
210	0.56392	0	0	0	0
220	0.5604	0	0	0	0
230	0.611902	0	0	3	0

FIGURE 5.12

Energy macro-model template

the bus wires that connect the masters and slaves. Energy macro-models were created for the first four components, with E_{WIRE} being calculated separately.

The energy macro-models for the bus matrix components are essentially of the form shown in Eq. (5.15). Leakage and clock energy (which are the major sources of independent energy consumption) are considered as part of the static energy coefficient α_0 for each of the components. Based on experimental results, an approximately linear relationship between cycle energy and macro-model variables was observed for the components. The energy models for each of the components are presented below.

5.2.1.1 *Input Stage*

Every master connected to a bus matrix has its own input stage that buffers address and control bits for a transaction, if a slave is busy. The input stage model can be expressed as:

$$E_{\text{INP}} = \alpha_{\text{inp0}} + \alpha_{\text{inp1}} \cdot \psi_{\text{load}} + \alpha_{\text{inp2}} \cdot \psi_{\text{desel}} + \alpha_{\text{inp3}} \cdot \psi_{\text{HDin}} + \alpha_{\text{inp4}} \cdot \psi_{\text{drive}} \quad (5.17)$$

where ψ_{load} and ψ_{drive} are control signals asserted when the register is loaded, and when the values are driven to the slave, respectively; ψ_{desel} is the control signal from the master to deselect the input stage when no transactions are being issued; and ψ_{HDin} is the Hamming distance of the address and control inputs to the register.