

Making ASIC power estimates before the design

A STRUCTURED APPROACH CAN GIVE YOU MEANINGFUL POWER ESTIMATES WHEN THERE'S STILL TIME TO INFLUENCE PROCESS AND ARCHITECTURE DECISIONS.

ASIC design teams face a growing problem. ASIC power-consumption estimates that take place before the design phase lack both scope and credibility. These shortcomings affect process, design, and IP (intellectual-property) selection. As the design complexity increases and power-consumption requirements tighten, this problem becomes more pervasive, often causing design rework, schedule slippage, higher NRE (nonrecurring-engineering) costs, and other challenges.

To address these issues, engineers should use a structured approach and a set of heuristic power-estimation rules for making effective predesign ASIC power estimates. Given that IC design flows and design requirements differ greatly, you may need engineering judgment and creativity to extend this approach across a range of design situations. In any event, these concepts should provide a good starting point. Companies that outsource portions of their ASIC design flows should engage early with their ASIC design partners to improve the quality of their predesign power estimates.

From an analytical viewpoint, generating power-consumption estimates should be straightforward. However, predesign power estimation is often given a low priority and may rely on the wrong mix of people to support the effort. In general, the team responsible for process, IP, and design selection should also own the predesign power estimates. This team will need augmentation if it is too distant from the design-implementation effort or if it has little hands-on power-estimation experience. For example, a software-centric architect plus a continually distracted manager will probably generate inadequate predesign ASIC power estimates, causing suboptimal process, design, and IP selection. That team needs designers with hands-on experience, and it may also need to include low-power-design experts from the ASIC design partner. The structured approach lends itself to distributing tasks and recombining results so that team members can share the workload.

GETTING STARTED

Power estimation involves dynamic power and static, or leakage, power. This article focuses on making approximate estimates; thus, it ignores various second-order effects, such as crossover current. To simplify the power-estimation process, perform the analyses of static and dynamic power estimations separately. If possible, prioritize the power-minimization requirements by determining the relative importance of static versus dynamic power consumption. For example, assume that

an ASIC design targets a battery-powered application in which it will be dynamically active 1% of the time and leaks current the other 99% of the time. In this case, minimizing the static power should take priority over minimizing the dynamic power.

IP selection and process selection are both critical but often seem like recursive problems. That is, IP selection drives process selection, and process selection drives IP selection. Fortunately, most applications should align with a requirement for low-power, general-purpose, or high-performance features. The application usually determines the choice of libraries and IP. The libraries and IP in turn help determine a target process. For example, most libraries and IP for portable devices should be available in low-power processes, and most libraries and IP for high-bandwidth networking should be available in high-performance processes. A check of library and IP availability will quickly help eliminate process choices and reduce the scope of the power-estimation effort before you start your design.

Once you have settled on a process node and type, you have bounded your operating voltage (Table 1). Note that the core voltage for low-power processes is 0.2V higher than the core voltage for general-purpose processes. So a low-power process has lower static power and higher dynamic power than the corresponding general-purpose process. This situation provides strong motivation for determining the relative importance of dynamic versus static power for your design.

USE SPREADSHEETS

Engineers prefer to use spreadsheets for early-phase power estimation because they are easy to use, help organize data, and support what-if design scenarios. Some companies may attempt to apply more elaborate approaches from the EDA industry. EDA tools generally provide their most effective results when you use them on well-defined designs. Before design start, the process, libraries, IP, and so forth remain undefined, which makes it difficult to estimate power with EDA tools. Using spreadsheets to estimate power consumption represents a rare design-community preference for software from Micro-

TABLE 1 NOMINAL ASIC-PROCESS CORE VOLTAGES

Process (nm)	Low-power process (V)	General-purpose process (V)
90	1.2	1
65	1.2	1
40	1.1	0.9

soft (www.microsoft.com), for example, rather than from the EDA industry.

MEMORY POWER ESTIMATION

Memory provides an excellent starting point for making initial power estimates. Complex designs may contain hundreds of types of memory, and the total memory power is often a substantial—if not the dominant—power component. You need neither a mature netlist nor powerful EDA tools to estimate memory power and study memory-power trade-offs. You need only the supplier's memory specifications or access to memory compilers to generate the memory specifications. Start with a sample of memory configurations for quick estimates and comparisons. Focus on memory configurations that consume the most power: those with high instance counts, that run at high frequencies, and that have large bit widths. As the design architecture matures, gradually include memories in the memory power estimates until you've covered all the memories in the design.

Contemporary memory compilers have various compilation options. Try the initial compilation runs without using power-saving features to obtain baseline data for supplier-to-supplier comparisons. You can then go back and explore power-saving features to help optimize the target implementations as the design matures. Track each memory type's width, depth, and power consumption. Also track the memory compiler's column-multiplexer option, which affects memory aspect ratios, performance, and power consumption. Use of inconsistent column-multiplexing options would invalidate data from comparative memory-compilation runs.

Vendor memory-power specifications can vary greatly, so designers need to work through the specifications to understand the conditions and make valid comparisons across memory suppliers. Memory specifications should break out dynamic power consumption employing read and write activity rates. However, the specification's footnotes may bury activity-rate information. If the activity rates remain unknown, ask the memory vendor about them. Also note that memory output ports usually drive relatively small loads, so dynamic power

IF A HIGH-THRESHOLD-VOLTAGE TRANSISTOR-BASED MEMORY IS TOO SLOW, YOU COULD USE A SIMILAR COMPILER TO GENERATE A SIMILAR MEMORY.

due to memory output loading should be far lower than the memory's net dynamic power consumption.

Data sheets don't provide exhaustive information that covers all cases of interest, so use scaling to generate design-specific data. When possible, you should also verify memory-power scaling against the actual memory-specification data to help validate the scaled results.

SCALING RULES

If you ignore second-order effects, you can assume that switching capacitance is constant across process and temperature. Thus, you can also assume that dynamic memory power—related to switching capacitance—is constant over process and temperature for rough power-estimation purposes. For example, if a high-threshold-voltage transistor-based memory is too slow, you could use a similar compiler to generate a similar memory based on faster, higher-leakage, low-threshold-voltage transistors. The high- and low-threshold-voltage versions of the memory should have equivalent layouts and dynamic power, but their leakage power differs greatly.

Memory dynamic power should scale as a function of voltage squared: $P=CV^2f$, where P is the power dissipated, C is the effective power-dissipation capacitance, V is the operating voltage, and f is the effective transition frequency. Because switching capacitance is fairly constant over process and temperature, for estimation purposes dynamic power becomes a function of just the voltage squared and the transition frequency. Fortunately, these variables are independent of each other, so you can focus on them separately. For example, if the memory vendor has provided best-case dynamic power at 1.32V, reflecting a 10% supply variation, but the design operates at or below 1.26V, reflecting a 5% supply variation, the dynamic-power-consumption scaling factor would be $(1.26)/(1.32)^2=0.91$.

To verify that the memory vendor's dynamic power scales as the square of the supply voltage, check the memory vendor's dynamic-power-spec data at typical and best-case conditions—say, 1.2 and 1.32V, respectively. Then, calculate the ra-

TABLE 2 SIMPLIFIED MEMORY-POWER SPREADSHEET

Memory	Word count	Bit width	Memory-instance count	Column multiplexer	Read activity (%)	Write activity (%)	Frequency (Hz)	Per-memory dynamic power (mW)	Per-memory static power (mW)	Dynamic power (mW)	Static power (mW)
One read/one write, high threshold voltage	7744	12	15	Eight	0.25	0.25	80	0.775	0.0034	11.64	0.051
One read/one write, low threshold voltage	4384	72	Two	Eight	0.25	0.25	270	9	0.025	18	0.050
One read/one write, high threshold voltage	5760	64	One	Eight	0.25	0.25	135	4.25	0.009	4.25	0.009
One read/one write, high threshold voltage	14,528	12	Two	16	0.25	0.25	80	0.925	0.005	1.85	0.010
Total										35.74	0.12

ratio of typical dynamic power to best-case dynamic power to see whether it yields $(1.20)^2/(1.32)^2$, or 0.826.

Memory dynamic power should scale linearly with frequency and activity rates: $P=CV^2fA$, where A is the activity rate, which can range from 0 to 100%. Many memory specifications provide simple power activity constants that you should multiply by frequency and activity rates. So you should include these power activ-

ity constants and memory frequencies in the memory-power spreadsheet to help automate the calculations. For example, assume that memory-vendor specs provide dynamic-power consumption at fixed activity rates, such as 20%, and the design runs at a maximum activity rate of 10%. You should scale down the 20% activity rate to derive power consumption with a 10% activity rate, using a scaling factor of 10%/20%, or 0.5.

Memory-read and memory-write activity rates should use different power activity factors. However, if read and write activity rates remain consistent, you can combine and scale them together. For example, assume a memory's read and write activity rates are both 15%, and you need to calculate what would happen with a 25% activity rate. You could combine these 15% read and write activity-rate results and scale them with the factor of 25%/15%, or 1.67. If the read and write activity rates differ, with a read activity rate of 25% and a write activity rate of 5%, for example, then don't combine the dynamic read and write power-consumption calculations. Instead, calculate the read-activity power separately from the write-activity power and then combine them.

Memory static power should remain fairly constant across activity rate and frequency. Thus, static-memory-power calculations should be independent of dynamic-memory-power calculations, and static-power calculations should use separate spreadsheet columns. Memory static power instead varies over process, voltage, and temperature. The highest leakage occurs with the best-case process and highest voltage corner at the highest temperature. These conditions differ from best-case timing conditions, which use the best-case process and voltage corner at the lowest temperature. It follows that you must carefully read the vendor's memory specs to ensure that the leakage data the vendor used in the power estimates reflects the highest leakage-power conditions, not the best-case timing.

Memory static power varies exponentially with temperature, so it doubles approximately every 15°C for 90-nm and larger processes. For newer, smaller processes, leakage tends to be less sensitive to temperature. You can study the effects of temperature on leakage by checking the memory-spec leakage data at, say, 25 and 85°C and calculating the temperature change for leakage to double. For example, $85-25=60^\circ\text{C}$ and $4 \times 15=60^\circ\text{C}$, so leakage should increase by approximately 2^4 , or 16 times, from 25 to 85°C. As another example, if the leakage specifications are at, say, 85°C but the design's maximum junction temperature is 100°C, the leakage-power estimate should be double the leakage power the vendor specifies at 85°C.

Table 2 shows a simplified memory-power spreadsheet. A more complete spreadsheet should include memory area, bit-count totals, read and write dynamic power per megahertz,

MAKE INITIAL STANDARD-CELL POWER ESTIMATES WITHOUT POWER-SAVING TECHNIQUES, SO YOU WILL HAVE BASELINE DATA.

and other variables. Also note that the memory in the second row with low threshold voltage has a much higher performance requirement than the other memories. To support higher performance, implement this memory with higher-leakage, low-threshold-voltage transistors; this instance accounts for 40% of the total static, or leakage, power. Efforts to minimize memory power should focus on memory configurations

consuming the most power, and your spreadsheet should make those configurations evident.

IP AND STANDARD-CELL POWER ESTIMATION

IP power estimation is similar to but less complex than memory power estimation. IP instance counts are usually much lower than memory instance counts, and there are usually fewer specialty-IP choices with fewer power-saving features, which simplifies power estimation. Other factors such as IP design risk, cost, and area may outweigh IP power consumption when you compare specialty IP from different suppliers. You should base your initial IP power estimation on the vendor's specifications. Enter the IP-power data into the power spreadsheets and separate the static- and dynamic-power components.

Make initial standard-cell power estimates without power-saving techniques, so you will have baseline data. Again, separate the static- and dynamic-power components. Predesign-phase estimates are difficult to make because you don't yet know the standard cell's instance count, multiple-threshold-voltage mix, routing capacitances, and clock-tree capacitances. In short, you don't know the capacitance in the CV^2f equation to make dynamic power estimates.

To work around not knowing the capacitance, you can scale the power results from a similar previous design, assuming that power data that includes routing capacitance is available. If you are targeting the same process node, scale for instance count and for frequency. If targeting a smaller process node, scale for instance count, frequency, voltage squared, and capacitance. Note that capacitance increases as spacing shrinks. For example, if gate-area scaling is 50%, use 60 to 90% for capacitance scaling.

An alternative approach to making dynamic power estimates involves scaling clock-tree power from a previous similar design. To do so, estimate the clock power per flip-flop that the previous design consumed. For example, assume a representative clock tree has 100,000 flip-flops and consumes 5 mW when including clock-tree routing capacitance. This scenario yields an average estimated power consumption of 0.05 μW per flip-flop, including clock-tree routing capacitance. Next, estimate clock-tree load counts in the new design. If necessary, scale for load capacitance due to library and process changes. To estimate flip-flop-load scaling factors, compare the clock-pin input capacitances of representative flip-flops in the previous design with similar flip-flops in the new design. Next, scale for frequency and the load count and then estimate the power per clock in the new design. Clock trees consume approximately one-half of the standard-cell dynamic

power in many ASICs, so double the clock-tree power to estimate total standard-cell power.

Estimating standard-cell static power is also problematic. One approach relies on scaling leakage-power results from a similar previous design, assuming that such data is available. If targeting the same process node and library, then scale the instance count to make the leakage-power estimate. You may also need to adjust for the relative mix of multiple-threshold-voltage cells. If targeting the same process node but with a different library, scale for instance count and for relative library leakage. You can base library-leakage scaling on the relative leakage of a small representative sample of standard cells. If targeting a different process node, scale for both instance count and relative library leakage.

The second approach for standard-cell leakage-power estimation relies on standard-cell library data. Select a representative sample of standard cells. You may need multiple samples of standard cells to account for different libraries and multiple-threshold-voltage-cell variations. Estimate the leakage for each sample of standard cells. Next, estimate the total instance count reflecting each sample of standard cells. Finally, scale the standard-cell samples to represent corresponding estimated design-instance counts and calculate the total static-power estimate.

For quick power estimates, start with a representative sample of key components. As the design matures, narrow your

Go to www.edn.com/ms4324 and click on Feedback Loop to post a comment on this article.

For more technical articles, go to www.edn.com/features.

process and IP choices and introduce more component data. Start with baseline data without power-saving features. Then, evaluate and work in appropriate power-saving features as the design progresses. To simplify the effort, try to separate the analysis of static and dynamic power. Also, focus on memory, IP, and standard-cell power separately. Recombine memory, IP, and standard-cell power estimates when necessary. IC design projects differ greatly, so you will need flexibility and

engineering judgment to handle design-specific power-estimation issues. **EDN**

AUTHOR'S BIOGRAPHY



Bob Eisenstadt is currently principal engineer at Alchip Technologies (Santa Clara, CA), a rapidly growing fabless-ASIC company. For the past 22 years, he has worked in a variety of ASIC design and consulting positions in Silicon Valley. He has extensive hands-on ASIC design experience in architecture, RTL (register-transfer-level) design, logic verification, timing analysis, physical design, and physical verification. In 2003, Eisenstadt co-founded Silicon Mosaic, where he developed and patented a generic power-gating product. He holds a bachelor's degree in electrical engineering from Cornell University (Ithaca, NY) and master's degrees in both electrical engineering and business administration from Santa Clara University (Santa Clara, CA).