Design Virtualization Technology For Low-Power ASICs

Chips can rarely meet their power budget if the whole chip is active all the time using a single library. So architectural design techniques have become very important. This paper discusses a variety of approaches for reducing power, including something new: design virtualization technology

Background

It was way back in 2001 that Pat Gelsinger, then CTO of Intel, pointed out that if we kept increasing clock rates that chips would have the power density of rocket nozzles and nuclear reactor cores. Ever since then power has been public enemy #1 in chip design.

In 2007 Apple announced the iPhone and the application processor inside it, and smartphones became one of the most intense battlegrounds for power. After all, the length of time that a battery lasts is much more visible to the consumer than the power dissipated by the chips in their wireless router. But routers are not immune to power either, at least at the datacenter level. Google, for example, has said that its electricity bill is larger than its salary expense. Maybe half that electricity goes into powering its datacenter routers and servers, and the other half goes into powering the air conditioning to get the heat dissipated out again.

There are two main classes of power, dynamic (or active) and standby (or leakage). With modern chip architectures both of these can be important at the same time, since blocks where the clock has been gated are only dissipating leakage power and blocks that are active are mainly dissipating dynamic power.

Leakage power is proportional to the supply voltage and dynamic power is proportional to the square of the power supply voltage. So the most effective way to reduce power is to lower the voltage. The technologists designing processes try to get this down as much as possible but for various technical reasons the voltage cannot be reduced as much at each node as used to be the case.

FinFET devices have higher drive current than planar devices, but perhaps the biggest attraction to them is lower leakage. At 20nm, transistors are not really on or off, but bright and dim, and a planar 16nm process is only going to be worse. FD-SOI also has much better leakage characteristics since the gate controls the channel so much better (and there are other power levers from the biasing available to reduce it further).

There is a sense in which all designs are low power. The power budget has to be met and it is usually a challenge. The lowest power chips for applications like hearing aids might have a power budget of a few milliwatts whereas a chip for a datacenter server might have a power budget of 150W. But both designs are power constrained and it is challenging to deliver the performance required (obviously very different in these two extreme examples) within the power envelope.

Power Reduction Options

Chips can rarely meet their power budget if the whole chip is active all the time using a single library. So architectural design techniques have become very important. The most important are:

Process design: most design groups don't get to play with this lever but every semiconductor technology development group is doing everything it can to keep leakage power, in particular, low, and to ensure that the design can run at as low a voltage as possible. This is usually limited by the fact that a process without working static RAM is not very useful.

Clock gating: the old rule is "never gate a clock," so a register that held a constant value was re-circulated through a multiplexor. Synthesis tools recognize this and similar structures and add a clock-gating cell. There are also more aggressive tools that can do sequential analysis and recognize that when the value of a register is not going to be used it does not matter if it is correct.

Library design: designing new standard cells and characterizing them is beyond the scope of most SoC designs, but the availability (or not) of power-optimized cells such as multi-bit flipflops can make the library choice very important.

Multiple threshold libraries: having two (or more) standard cell libraries can be useful. For example: a high-performance, high-power, high-leakage one and a low-performance, low-power, low-leakage one. Synthesis tools will switch cells in and out so that only the most critical timing paths use high-performance libraries.

Gate-biasing: at the layout (foundry) level it may be possible to further optimize by making transistor lengths longer for transistors where the spec performance is not required, thus reducing leakage, or, perhaps, shaving down the lengths of the highest-performance transistors.

Voltage islands: different areas of the chip may require different levels of performance and there is scope for running these different areas at different voltages, only using higher voltages on the high-performance parts of the chip that require it.

Turn off the clock: the clock on a big SoC can dissipate 30-50 percent of the power, so if blocks are inactive a lot of power can be saved by simply gating the clock to the entire block. Care needs to be taken when turning the clock back on again since this can generate large transient currents that must be analyzed. This only affects dynamic power; since the block is still powered it still leaks.

Turn off the power: if a block is not required for a long time (at least in SoC years) then the power can be turned off completely. This saves both dynamic and leakage power. There are sometimes two levels of turning off the power, one where some sort of retention scheme is used to ensure that registers retain their values even while the block is otherwise off, and a completely quiescent block. Care needs to be taken when turning the power back on again. For normal operation very large transistors are required to transfer power into the block.

But if the block is off, these transistors cannot simply be turned on since the inrush current will cause so much voltage drop that the rest of the chip will probably not function. Instead, the block needs to be brought up to operating voltage slowly before turning on the main power gating transistors.

This is rather like a ship in a canal-lock. You can't just close the lower gate and open the upper. First you need to bring the water level in the lock up so that when the upper gates are opened the lock itself is already at the level of the upper part of the canal.

Dynamic voltage and frequency scaling (DVFS): a block like a microprocessor may be able to be run at varying frequencies depending on the computation load. This can be done simply by changing the clock frequency, but for the greatest power savings the voltage should also be reduced as much as possible: a lower frequency does not require such a high supply voltage. This is a complex technique to implement since the frequency must be reduced before the voltage when slowing down, and the voltage must be increased before the frequency when speeding up.

Race-to-halt: running a microprocessor slowly is not always the most power efficient thing to do since it may also require a large part of the SoC to be powered up whenever the processor is running. An alternative is race-to-halt, whereby the microprocessor is run at maximum performance until its workload is complete, and then the whole microprocessor subsystem can be turned off (through clock gating or power gating).

Tighten the operating conditions: libraries are often characterized from -25°C. A chip for a data-server application, for example, is never going to operate below, for example, 20°C, so characterizing at 0°C should still give plenty of margin and allow the synthesis tool to pick fewer high-performance cells. Similarly, a chip that is going into a cell phone is never going to run at 125°C, so lowering the upper temperature will provide more margin for setup/hold fixing.

Tighten the process window: the worst-case (highest) power dissipation condition typically occurs at one of the process corners. Pulling those corners in a little can result in big power savings at the possible loss of some yield. This is especially attractive in a mature process since the process corners were often set conservatively before the process was ramped to volume and a lot of parametric data became available. The traditional 3σ window used is extremely conservative and reducing it to a 2σ window can make a huge difference to power at the cost of a maximum of 5 percent yield loss.

Tighten the voltage window: usually people assume a voltage regulator is within 5 percent but this may be very conservative. There are other areas that may be overly conservative, such as the assumption about IR drop from the power pins to the core, where being less conservative (especially if an accurate analysis is done) can lead to power savings.

Worry a lot about memories: many SoCs have large amounts of memory on them, which can be responsible for a disproportionate share of the power. It is not often possible to power memories down completely because, by definition, we usually want them to remember stuff. But often they have lower-power standby modes whereby they retain their contents but can neither be read nor written.

Run memory cores at a higher voltage than the rest of the chip: often the limiting factor on how low the voltage can go is the requirement that static memories work. A dual rail solution that uses a higher supply voltage for the core while running the rest of the chip at a lower voltage can have a big impact.

Custom bit cell: the bit cell of a memory is almost always provided by the foundry. Designing a custom bit cell can allow the voltage to be lowered further if other aspects of the characterization can be tightened. This is clearly a "for experts only" technique beyond the scope of most SoC design groups, but sometimes it is necessary for designs with extreme requirements.

Some of these are architectural techniques that really need to be incorporated into the design from the very beginning as part of the feasibility study of the SoC.

For example, powering down a block will almost always be a good thing if it is dormant for long periods and the system can cope with the latency involved in turning it back on.

But against that needs to be set the additional complexity of layout (you can only power down a block on a contiguous area of the chip), analysis (especially IR and signal integrity analysis) and the associated schedule impact. Some of these techniques, such as DVFS or designing a custom bit cell for a memory, are almost in the "are you crazy?" league.

But there are other more difficult decisions:

- Which process should be used?
- How low can the voltage go?
- Which standard cell libraries should be used?
- Which memories should be used?
- What process corners should be used?
- What voltage regulator accuracy should be used? IR drop? Timing windows?
- Can the temperature range be reduced at the lower or upper end (or both)?

Design Virtualization

eSilicon's design virtualization technology adds a layer between the design itself and the actual silicon, allowing many of these decisions to be analyzed in detail, facilitating optimal choices. This can happen before the design is started, when obviously only the roughest of detail is available, or when a design is close to completion, when very accurate numbers should be available.

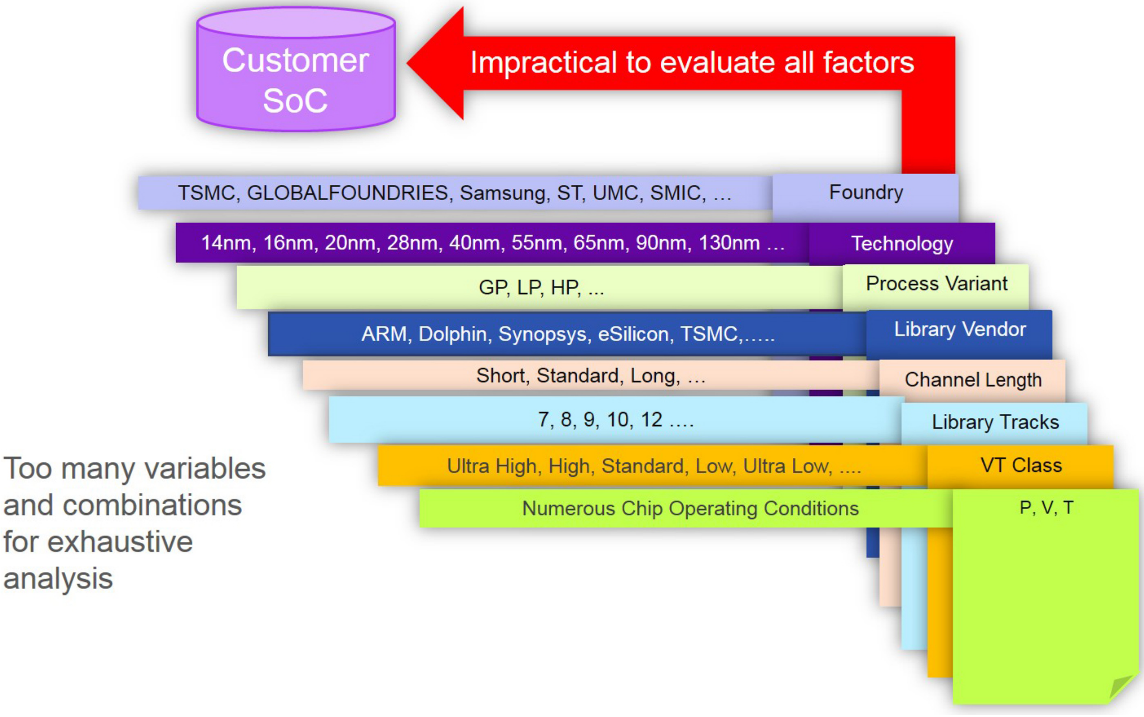
There are far too many choices to evaluate by hand and there is too much data that is simply inaccessible to any particular design group, which typically will only have access to very generic data, along with detailed data on their own designs.

Design virtualization handles this problem by using Big Data and Machine Learning techniques. Another issue it addresses is how a design will perform with a particular set of process options. It seems obvious that any low-power design should be done in a low-power process. But, in fact, all designs are low-power and the reality is that if a design has a very high duty cycle then the high-performance process might be better. On the other hand, if a design is almost always idle (think of the camera chip in a phone) then leakage will dominate completely and the low-power process will be the best choice. Somewhere in between there is a non-obvious crossover point.

Today, design virtualization technology is provided by eSilicon as a service. It makes doing a what-if comparison very simple and fast compared with the alternative of doing the design twice, no matter how many corners were cut on the duplicate design. By doing additional characterization above and beyond the usual corners, the technology can predict how a design will behave at obscure "corners" such as 93°C at VDD+3%.

Accurate analysis allows designers to be much more aggressive about signing off for power with confidence. As an example, for a

Figure 1: The number of options to choose from in a new ASIC design can be overwheliming


Customer SoC
Impractical to evaluate all factors
TSMC, GLOBALFOUNDRIES, Samsung, ST, UMC, SMIC, …
Foundry
14nm, 16nm, 20nm, 28nm, 40nm, 55nm, 65nm, 90nm, 130nm …
Technology
GP, LP, HP, …
Process Variant
ARM, Dolphin, Synopsys, eSilicon, TSMC,…..
Library Vendor
Short, Standard, Long, …
Channel Length
7, 8, 9, 10, 12 ….
Library Tracks
Ultra High, High, Standard, Low, Ultra Low, ….
VT Class
Numerous Chip Operating Conditions
P, V, T
Too many variables and combinations for exhaustive analysis

28nm design, the following might turn out to be usable with a big impact on both dynamic and static power:

Table 1: Reducing dynamic and static power

Parameter	Standard Signoff	Aggressive Signoff
VDD	0.9V	0.864V
IR drop	45mV	27mV
Voltage regulator tolerance	45mV	27mV
Timing signoff	SS 0.81V -40°C	SS 0.81V 0C
Power signoff	FF 0.945V 125°	FFG 0.891V 105°C

Design Example

Achieving aggressive power reduction goals is a multi-faceted problem. An effective approach touches IP, process and design methodology. eSilicon's design virtualization technology provides an environment that brings all of these elements together.

Rather than give the marketing pitch, I think it is good to actually show a real-world example of the technology in action on a real design. eSilicon worked with a customer on a networking design. When the design was essentially complete the first power estimate came up at 130W, which was far above the power budget of 75W. So eSilicon worked with the customer through the following steps:

Where is the power coming from? Ninety-five percent of the power turned out to be coming from a single 450MB memory.

Can we customize the memory? Yes. So eSilicon did that using an off-the-shelf memory from their extensive portfolio of internally developed memory IP. They then removed all the peripheral logic that supports options not required for this particular design. There were also device swaps in the periphery by using libraries with multiple thresholds.

This got the power down to 90W, but the customer target was lower at 75W. eSilicon's design virtualization technology analyzed the design. By applying low-power techniques such as lowering the core voltage a tiny bit, using more multi-Vt libraries and so on, they achieved 75W. The chip was three days away from its scheduled tapeout. At this stage life looked good.

Marketing came back and said the power budget had to be 35W. eSilicon fired up their design virtualization technology again. If they got aggressive at voltage, temperature and process corners, could they get there? What about frequency? Was there any flexibility to reduce that and still meet the performance requirements?

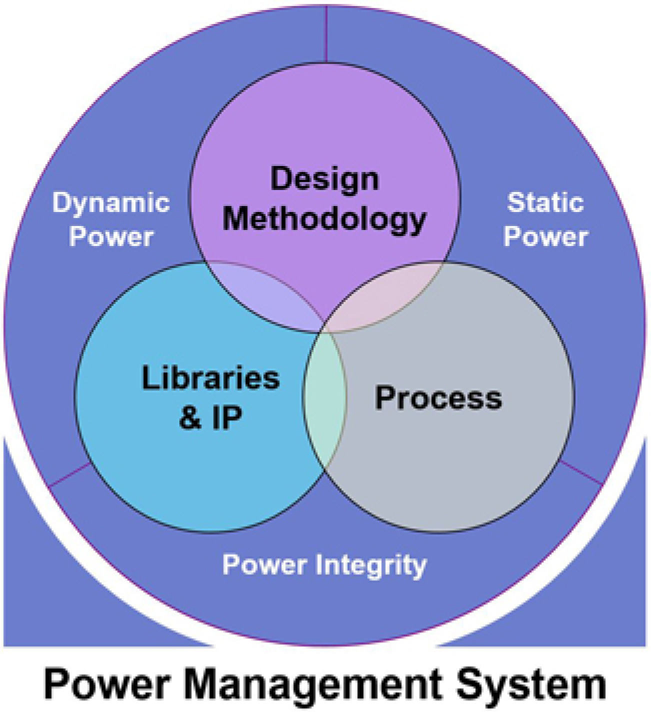
It turned out they needed to do all four:

- Voltage down two percent.
- Tighter process window, eating a tiny potential yield loss, temperature maximum of 105°C.
- Frequency from 500MHz down to 400MHz.
- Power at 35W.

Tapeout on schedule three days later. Success.

This success story would have been significantly more difficult and time consuming without eSilicon's design virtualization technology to guide the way with a detailed analysis of multiple scenarios.

Figure 2: Low-power ASIC design requires a balance across many parameters



For more information on design virtualization, contact info@esilicon.com.