In Chapter 3, we observed that the optimal operation point in Energy–Delay space is a strong function of the activity — or, in other words, the operation mode of the circuit — and that there exists an optimal ratio between dynamic and static power dissipation. One special case is when there is no computational activity going on at all, that is, the standby mode. In an ideal world, this would mean that the dynamic power consumption should be zero or very small. Moreover (given the constant ratio), static power dissipation should be eliminated as well. Although the former can be achieved through careful management, the latter is becoming harder with advanced technology scaling. When all transistors are leaky, completely turning off a module is hard. In this chapter, we discuss a number of circuit and system techniques to keep both dynamic and static power in standby to an absolute minimum. As standby power is the main concern in memories (and as memories are somewhat special anyhow), we have relegated the discussion on them to Chapter 9.
Chapter Outline

- Why Sleep Mode Management?
- Dynamic power in standby
  - Clock gating
- Static power in standby
  - Transistor sizing
  - Power gating
  - Body biasing
  - Supply voltage ramping

Slide 8.2
We start the chapter with a discussion on the growing importance of reducing standby power. Next, we analyze what it takes to reduce dynamic power in standby to an absolute minimum. The bulk of the chapter is devoted to the main challenge; that is, the elimination (or at least, minimization) of leakage during standby. Finally, some future perspectives are offered.

Slide 8.3
With the advent of mobile applications, the importance of standby modes has become more pronounced, as it was realized that standby operation consumes a large part of the overall energy budget. In fact, a majority of applications tend to perform in a bursty fashion – that is, they exhibit short periods of intense activity interspersed with long intervals of no or not much activity. This is the case even in more traditional product lines such as microprocessors. Common sense dictates that modules or processors not performing any task should consume zero dynamic and also (preferably) zero static power.
When power became an issue, this problem was quickly corrected as shown in the traces for the Pentium-2.

Slide 8.5
The main source of dynamic energy consumption in standby mode is the clock. Keeping the clock connected to the flip-flops of an idle module not only adds to the clock loading, but may cause spurious activity in the logic. In fact, as the data that is applied under those conditions is actually quite random, activity may be maximized as we have discussed earlier. This wasteful bit-flipping is avoided by two design interventions:

- Disconnect the clock from the flip-flops in the idle module through clock gating.
- Ensure that the inputs to the idle logic are kept stable. Even without a clock, changes at the inputs of a combinational block cause activity.

Clock gating a complete module (rather than a set of gates) makes the task a lot easier. However, deciding whether a module, or a collection of modules, is idle may not always be straightforward. Though sometimes it is quite obvious from the register-transfer level (RTL) code, normally it requires an understanding of the operational system modes. Also, clock gating can be more effective if modules that are idle simultaneously are grouped. What this basically says is that standby-power management plays at all levels of the design hierarchy.
logic determines which data path units are needed for its execution, and subsequently set their Enable signals to 1.

As the inputs of the logic module are connected to the register file, they remain stable as long as the clock is disabled. In the case that the inputs are directly connected to a shared bus, extra gates must be inserted to isolate the logic.

Observe that the gated clock signal suffers an additional gate delay, and hence increases the skew. Depending upon the time in the design process it is inserted, we must ensure that this extra delay does not upset any critical set-up and hold-time constraints.

**Slide 8.6**

One possible way of implementing clock gating is shown in this slide. The clock to the register files at the inputs of an unused module is turned on or off using an extra AND gate controlled by an Enable signal. This signal is either introduced explicitly by the system- or RTL-designer, or generated automatically by the clock synthesis tools. Take for instance the case of a simple microprocessor. Given an instruction loaded in the instruction register (IR), the decoding logic determines which data path units are needed for its execution, and subsequently set their Enable signals to 1.

As the inputs of the logic module are connected to the register file, they remain stable as long as the clock is disabled. In the case that the inputs are directly connected to a shared bus, extra gates must be inserted to isolate the logic.

Observe that the gated clock signal suffers an additional gate delay, and hence increases the skew. Depending upon the time in the design process it is inserted, we must ensure that this extra delay does not upset any critical set-up and hold-time constraints.

**Slide 8.7**

There is no doubt that clock gating is a truly effective means of reducing standby dynamic power. This is illustrated numerically with the example of an MPEG4 decoder [Ohashi’02]. Gating 90% of the flip-flops results in a straight 70% standby power reduction. This clearly indicates that there is NO excuse for not using clock gating in today’s power-constrained designs.
Clock Gating

- Challenges to skew management and clock distribution (load on clock network varies dynamically)
- Fortunately state-of-the-art design tools are starting to do a better job
  - For example, physically aware clock gating inserts gaters in clock tree based on timing constraints and physical layout

Power savings

![Clock Gating Diagram]

Yet, as mentioned, these gains do not come for free, and present an extra burden on the designers of the clock distribution network. In addition to the extra delay of the gating devices, clock gating causes the load on the clock network to vary dynamically, which introduces another source of clock noise into the system.

Let us, for instance, explore some different options on where to introduce the gating devices in the clock-tree hierarchy.

One possible solution is to keep the gaters close to the registers. This allows for a fine-grain control on what to turn off and when. It comes at the expense of a more complex skew control and extra area. Another option is to move the gating devices higher up in the tree, which has the added advantage that the clock distribution network of the sub-tree is turned off as well – leading to some potentially large power savings. This comes at the expense of a coarser control granularity, which means that modules cannot be turned off as often.

Given the complexity of this task, it is fortunate that state-of-the-art clock synthesis tools have become more adept in managing the skew in the presence of clock gating. This will be discussed in more detail later, in the chapter on design methodology for power (Chapter 12).

Clock Hierarchy and Clock Gating

Example: Clock distribution of dual-core Intel Montecito processor

![Clock Hierarchy Diagram]

© IEEE 2005

[Ref: T. Fischer, ISSCC’05]

Slide 8.8

However effective these tools may be, it will be some time before they can handle the full complexity of the clock network of a modern microprocessor design. A bird’s-eye view on the clock network of the dual-core Intel Montecito processor is shown here. Each core is allowed to run on variable frequencies (more about this in Chapter 10, when we discuss runtime optimization). The digital frequency dividers (DFDs) translate
the central clock to the frequency expected for the different clock zones. The downstream clock network employs both active deskew (in the second-level clock buffers or SLCBs, and in the regional active deskew or RAD) and fixed deskew, tuned via scan (using the clock vernier devices or CVDs). The latter allow for final fine-tuning. Gaters provide the final stage of the network, enabling power saving and pulse shaping. A total of 7536 of those are distributed throughout the chip. Clock gating clearly has not simplified the job of the high-performance designer!

Slide 8.10
The introduction of clock gating succeeds in virtually eliminating the dynamic power dissipation of the computational modules during standby. However, although the ends of the clock tree have been disconnected, the root is still active and continues to consume power. Further power reductions would require that the complete clock distribution network and even the clock generator (which typically includes a crystal-driven oscillator and a phase-locked loop) are put to sleep. Although the latter can be turned off quite quickly, bringing them back into operation takes a considerable amount of time, and hence only makes sense if the standby mode is expected to last for considerable time.

Many processors and SoCs hence feature a variety of standby (or sleep) modes, with the state of the clock network as the main differentiator. Options are:

- Just clock gating
- Disabling the clock distribution network
- Turning off the clock driver (and the phase-locked loop)
- Turning off the clock completely.

In the latter case, only a wake-up circuit is kept alive, and the standby power drops to the microwatt range. Companies use different names for the various modes, with sleep mode typically reserved for the mode where the clock driver is turned off. It may take tens of clock-cycles to bring a processor back to operation from sleep mode.

Slide 8.11
The choice of the standby modes can be an important differentiator, as shown in this slide for a number of early-day low-power microprocessors. The Motorola PowerPC 603 supported four different operation modes, ranging from active, to doze (clocks still running to most units), nap (clock only to a timer unit), and sleep (clock completely shut off). The MIPS on the other hand did not support a full sleep mode, leading to substantially larger power dissipation in standby mode.

The MSP430™ microcontroller from Texas Instruments shows the state-of-the-art of standby management. Using multiple on-chip clock generators, the processor (which is actively used in
low-duty-cycle power-sensitive control applications) can go from standby (1 μA) to active mode (250 μA) in 1 μs. This rapid turnaround helps to keep the processor in standby longer, and makes it attractive to go into standby more often.

Slide 8.12
From the previous slides, a new version of our classic E–D trade-off curve emerges. The metrics to be traded off here are standby power versus wake-up delay.
Also the Case for Peripheral Devices

**Hard disk**

<table>
<thead>
<tr>
<th></th>
<th>P(_{\text{sleep}}) W</th>
<th>P(_{\text{active}}) W</th>
<th>T(_{\text{sleep}}) sec</th>
<th>T(_{\text{active}}) sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM</td>
<td>0.75</td>
<td>3.48</td>
<td>0.51</td>
<td>6.97</td>
</tr>
<tr>
<td>Fujitsu</td>
<td>0.13</td>
<td>0.95</td>
<td>0.67</td>
<td>1.61</td>
</tr>
</tbody>
</table>

**Wireless LAN Card**

<table>
<thead>
<tr>
<th></th>
<th>TX</th>
<th>RX</th>
<th>Doze</th>
<th>Off</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power</td>
<td>1.65 W</td>
<td>1.4 W</td>
<td>0.045 W</td>
<td>0 W</td>
</tr>
<tr>
<td>Transitions</td>
<td>To Off: 82 ms</td>
<td>To Doze: 34 ms</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[Ref: T. Simunic, Kluwer'02]

Although standby modes are most often quoted for processors, they make just as much (if not more) sense for peripheral devices. Disks, wired and wireless interfaces, and input/output devices all operate in a bursty fashion. For instance, a mouse is in standby most of the time, and even when operational, data is only transmitted periodically. Clock gating and the support of different standby modes are hence essential. In this slide, the measured power levels and the transition times for two such peripheral devices are shown. Clearly the timing overhead associated with the wake-up from the standby mode cannot be ignored in each of these. Cutting down that time is crucial if standby is to be used more effectively.

**The Leakage Challenge – Power in Standby**

- With clock gating employed in most designs, leakage power has become the dominant standby power source
- With no activity in module, leakage power should be minimized as well
  - Remember constant ratio between dynamic and static power …
- Challenge – how to disable unit most effectively given that no ideal switches are available

**Slide 8.13**

Given the effectiveness of clock gating, there is little excuse for dynamic power dissipation in standby. Eliminating or drastically reducing standby currents is a lot more problematic. The main challenge is that contemporary CMOS processes do not feature a transistor that can be turned off completely.
Standby Static Power Reduction Approaches

- Transistor stacking
- Power gating
- Body biasing
- Supply voltage ramping

A standby leakage control technique must be such that it has minimal impact on the normal operation of the circuit, both from a functional and performance perspective. Lacking a perfect switch, only two leakage-reduction techniques are left to the designer: increase the resistance in the leakage path, or reduce the voltage over that path. As the latter is harder to accomplish—you need either a variable or multiple supply voltages—most of the techniques presented in this chapter fall in the former category.

Transistor Stacking

- Off-current reduced in complex gates (see leakage power reduction @ design time)
- Some input patterns more effective than others in reducing leakage
- Effective standby power reduction strategy:
  - Select input pattern that minimizes leakage current of combinational logic module
  - Force inputs of module to correspond to that pattern during standby
- Pros: Little overhead, fast transition
- Con: Limited effectiveness

In Chapter 4 we established that the stacking of transistors has a super-linear leakage reduction effect. Hence, it pays to ensure that the stacking effect is maximized in standby. For each gate, an optimal input pattern can be determined. To get the maximum effect, one has to control the inputs of each gate individually, which is unfortunately not an option. Only the primary inputs of a combinational block are controllable. Hence, the challenge is to find the primary input pattern that minimizes the leakage of the complete block. Even though stacking has a limited impact on the leakage, the advantage is that it virtually comes for free, and that it has a negligible impact on performance.
Transistor Stacking

Combinational Module

Latches

Clk

Standby

Latches

Slide 8.17
Standby leakage control using the stack effect requires only one real modification to the circuitry: all input latches or registers have to be presetable (either to the “0” or to the “1” state). This slide shows how this modification can be accomplished with only a minor impact on performance. Once the logic topology of a module is known, computer-aided design (CAD) tools can easily determine the optimal input pattern, and the corresponding latches can be inserted into the logic design.

Forced Transistor Stacking

Useful for reducing leakage in non-critical shallow gates (in addition to high $V_{TH}$)

[Ref: S. Narendra, ISLPED’01]

Slide 8.18
Even when the technology-mapping phase of the logical-synthesis process is acutely aware of the stacking opportunity, it is unavoidable that some gates in the module end up with small fan-in. An inverter here or there is hard to avoid. And these simple gates contribute largely to the leakage. This can be remedied through the use of forced stacking, which replaces a transistor in a shallow stack by a pair (maintaining the same input loading). Although this transistor doubling, by necessity, impacts the performance of the gate – and hence should only be used in non-critical paths – the leakage reduction is substantial. This is perfectly illustrated by the leakage current (i.e., standby power) versus delay plots, shown on the slide for the cases of high- and low-threshold transistors. The advantage of forced stacking is that it can be fully automated.

Observe that this slide introduces another important trade-off metric: standby power versus active delay.
The ideal way to eliminate leakage current is to just disconnect the module from the supply rails – that is, if we could have perfect on–off switches available. The next best option is to use switches acting as “large resistors” between the “virtual” supply rails of the module and the global supply rails. Depending upon their position, those switches are called “headers” or “footers”, connecting to $V_{DD}$ or ground, respectively. This power-gating technique performs the best when the technology supports both high- and low-threshold transistors. The latter can be used for the logic, ensuring the best possible performance, whereas the others are very effective as power-gating devices. When multiple thresholds are used, the power-gating approach is often called MTCMOS.

---

**Slide 8.20**

The headers/footers add resistance to the leakage path during standby. In addition, they also introduce a stacking effect, which increases the threshold of the transistors in the stack. The combination of resistance and threshold increase is what causes the large reduction in leakage current.
Obviously, introducing an extra transistor in the charge and discharge paths of a gate comes with a performance penalty, the effects of which we would like to mitigate as much as possible. In principle, it is sufficient to insert only a single transistor (either footer or header) for leakage reduction. The addition of the second switch, though far less dramatic in leakage reduction, ensures that the stacking effect is exploited independent of the input patterns. If one chooses a single power-gating device, the NMOS footer is the preferred option, because its on-resistance is smaller for the same transistor width. It can hence be sized smaller than its PMOS counterpart. This is the approach that is followed in a majority of today’s power-conscious IC designs.

A number of modifications to the standard power-gating techniques can be envisioned, producing even larger leakage reductions, or reducing the performance penalty. The “boosted-gate” approach raises the gate voltage of the footer (header) transistors above the supply voltage, effectively decreasing their resistance. This technique is only applicable when the technology allows for high voltages to be applied to the gate. This may even require the use of thick-oxide transistors. Some CMOS processes make these available to allow for the design of voltage-converting input and output pads (Note: the core of a chip often operates at a supply voltage lower than the board-level signals to reduce power dissipation).
Other Option: Boosted-Sleep MOS
(also called Super-Cutoff CMOS or SCCMOS)

Other Option: Boosted-Sleep MOS
(also called Super-Cutoff CMOS or SCCMOS)

Leak cut-off Switch (LS)
- normal (or high) $V_{TH}$
- normal $T_{OX}$

Area-efficient

Virtual GND

<Standby><Active>
$V_{DD}$
0 V
$V_{boost}$

[Ref: T. Inukai, CICC’00]

Slide 8.23
The reverse is also possible. Instead of using a high-$V_{TH}$ device, the sleeper transistor can be implemented with a low-$V_{TH}$ device, leading to better performance. To reduce the leakage in standby, the gate of the sleeper is reverse biased. Similar to the “boosted-gate” technique, this requires a separate supply rail. Be aware that this increases the latch-up danger.

Virtual Supplies

Active Mode

Standby Mode

Noise on virtual supplies

Virtual supply collapse

[Ref: J. Tschanz, JSSC’03]

Virtual Supplies

Slide 8.24
It is worth observing what happens with the virtual supplies in active and sleep modes. The extra resistance on the supply rail not only impacts performance, but also introduces extra $IR$-induced supply noise – impacting the signal integrity. During standby mode, the virtual supply rails start drifting, and ultimately converge to voltage levels determined by the resistive divider formed by the on and off transistors in the stack. The conversion process is not immediate though, and is determined by the leakage rates.

Slide 8.25
Reaching the standby operation mode is hence not immediate. This poses some interesting questions on where to put most of the decoupling capacitance (decap): on the chip supply rails, or on the virtual rails? The former has the advantage that relatively lower capacitance has to be (dis)charged when switching modes, leading to faster convergence and smaller overhead. The cost and overhead of going to standby mode is smaller. Also, the energy overhead for charging and discharging the decoupling capacitance is avoided. This approach
also has some important disadvantages: (1) the virtual supplies are more prone to noise, and (2) the gate-oxide capacitance that serves as decap stays under full voltage stress, and keeps contributing gate leakage current even in standby (Note that on-chip decoupling capacitance is often realized using huge transistors with their sources and drains short-circuited). Putting the decap on the chip supply rails hence is the preferred option if standby mode is most often invoked for a short time. The “decap on virtual supply” works best for long, infrequent standby invocations.

Slide 8.26
This trade-off is illustrated in the simulation of the virtual supply rails. After 10 ms, the leakage power of the “no decap on virtual rails” scenario has dropped by 90%. It takes 10 times as long for the “decap on virtual rails” to reach the same level of effectiveness.

Slide 8.27
As mentioned earlier, the sleep transistor does not come for free, as it impacts the performance of the module in active mode, introduces supply noise, and costs extra area. To minimize the area, a single switch is most often shared over a set of gates. An important question hence is how to size the transistor: making it wider minimizes performance impact and noise, but costs area. A typical target for sleep transistor sizing is to ensure that the extra ripple on the supply rail is smaller than 5% of the full swing.
If the designer has access to power distribution analysis and optimization tools, sizing of the sleep transistors can be done automatically—as we will discuss in Chapter 12. If not, it is up to her to determine the peak current of the module through simulations (or estimations), and size the transistor such that the voltage drop over the switch is no larger than the allowable 5%.

Slide 8.28
The table on this slide compares the effectiveness of the different power-gating approaches. In the MTCMOS approach, a high-\(V_{TH}\) device is used for the sleeper. To support the necessary current, the transistor must be quite large. When a low-\(V_{TH}\) transistor is used, the area overhead is a lot smaller at the expense of increased leakage. The boosted-sleep mode combines the best of both, that is small transistor width and low leakage, at the expense of an extra supply rail. The transistors were sized such that the supply bounce for each of them is approximately the same.

Slide 8.29
The attentive reader must already have wondered about one important negative effect of power gating: when we disconnect the supply rails, all data stored in the latches, registers and memories of the module ultimately are lost. This sometimes is not a problem, especially when the processor always restarts from the same initial state—that is, all intermediate states can be forgotten. More often, the processor is expected to remember some part of its prior history, and rebooting from scratch after every sleep period is not an option. This can be dealt with in a number of ways:

- All essential-states are copied to memory with data retention before going to sleep, and is reloaded upon restart. Everything in the scratch-pad memory can be forgotten. The extra time for copying and reloading adds to the start-up delay.
The essential memory in the module is not powered down, but put in data retention mode. This approach increases the standby power, but minimizes the start-up and power-down timing overhead. We will talk more about memory retention in the next chapter.

Slide 8.30
The latter approach has the smallest granularity of control, but yields the smallest reduction in standby leakage, as all registers are still powered up. Also, the active performance of the latch/register may only be minimally impacted.

An example of a master slave register that combines high-speed active-mode performance with low-leakage data retention is shown in this slide. High-$V_{TH}$ devices are used for all transistors except for shaded transistors or gates that are critical. Black-shaded devices use low-$V_{TH}$ transistors. All others are high-$V_{TH}$.

[Ref: S. Mutoh, JSSC'95]

Slide 8.31
The above represents only a single example of a data retention latch. Many others have been perceived since then.

A very different option for state retention is to ensure that the standby voltage over the registers does not drop below the retention voltage (i.e., the minimum voltage at which the register or latch still reliably stores data). This is, for instance, accomplished by setting the standby voltage to $V_{DD} - V_D$ (where $V_D$ is the voltage over a reverse-biased diode), or by connecting it to a separate supply rail called $V_{retain}$. The former approach has the advantage that no extra supply is needed, whereas the latter allows for careful selection of the retention voltage so that leakage is minimized.
while retaining reliable storage. Both cases come with the penalty that the leakage through the logic may be higher than what would be obtained by simple power gating. The advantage is that the designer does not have to worry about state retention.

The topic of retention voltages and what determines their value is discussed in more detail in the next chapter on memory standby (Chapter 9).

Slide 8.32
To conclude the discussion on power gating, it is worth asking ourselves how this impacts layout strategy and how much area overhead this brings with it. Fortunately power switches can be introduced in contemporary standard layout tools with only minor variations.

In a traditional standard-cell design strategy, it is standard practice to introduce “strapper” cells at regular intervals, which connect the \( V_{DD} \) and \( GND \) wires in the cells to the global power distribution network. These cells can easily be modified to include header and footer switches of the appropriate sizes. Actually, quite often one can determine the size of the switches based on the number of the cells they are feeding in a row.
The area overhead of the power-gating approach was quantified in a study performed at Intel in 2003 [Ref: J. Tschanz, ISSCC’03], which compared the effectiveness of various leakage control strategies for the same design (a high-speed ALU). Both footers and headers were used, and all sleep transistors were implemented using low-threshold transistors to minimize the impact on performance. It was found that the area overhead of the power gating was 6% for the PMOS devices, and 3% for the NMOS footers. We will come back to the same study in a couple of slides.

Dynamic Body Biasing

- Increase thresholds of transistors during sleep using reverse body biasing
  - Can be combined with forward body biasing in active mode
- No delay penalty

But

- Requires triple-well technology
- Limited range of threshold adjustments (<100 mV)
  - Not improving with technology scaling
- Limited leakage reduction (<10x)
- Energy cost of charging/discharging the substrate capacitance

An alternative to the power-gating approach is to decrease the leakage current by increasing the thresholds of the transistors. Indeed, every transistor has a fourth terminal, which can be used to increase the threshold voltage through reverse biasing. Recall that a linear change in threshold voltage translates into an exponential change in leakage current. Even better, this approach can also be used to decrease the transistor threshold in active mode through forward biasing! The alluring feature of dynamic biasing of the transistor is that it does not come with a performance penalty, and it does not change the circuit topology. The only drawback seems to be the need for a triple-well technology if we want to control both NMOS and PMOS transistors.

Although all this looks very attractive at a first glance, there are some other negatives that cannot be ignored. The range of the threshold control through dynamic biasing is limited, and, as we established in Chapter 2, it is rapidly decreasing with the scaling of the technology below 100 nm. Hence the effectiveness of the technique is quite small in nano-meter technologies, and will not get better in the future unless novel devices with much better threshold control emerge (for instance, the dual-gate transistors we briefly introduced in Chapter 2). Finally, changing the
back-gate bias of the transistors requires the charging or discharging of the well capacitance, which adds a sizable amount of overhead energy and time.

**Slide 8.35**
The concept of dynamic body biasing (DBB), as first introduced by Seta et al. in 1995, is illustrated pictorially in this slide. Obviously the approach needs some extra supply voltages that must be distributed over the chip. Fortunately, these extra supplies have to carry only little continuous current, and can be generated using simple on-chip voltage converters.

The technique of dynamic body biasing is by no means new, as it has been applied in memory designs for quite some time. Yet, it is only with leakage power becoming an important topic that it is being applied to computational modules. The attentive reader probably realizes that this technique has more to offer than just leakage management. It can, for instance, also be used for the compensation of threshold variations. To hear more about this, you have to wait for the later chapters.

**Slide 8.36**
Though the adoption of DBB requires little changes in the computational modules, it takes some extra circuitry to facilitate the switching between the various biasing levels, which may extend above or below the standard voltage rails. Adapting the sleep control signals (CE) to the appropriate levels requires a set of level converters, whose outputs in turn are used to switch the well voltages. The resulting voltage waveforms, as recorded in [Seta95], are shown on the slide as well. Observe that in this early incarnation of the DBB approach it takes approximately the same time to charge and discharge the wells – for a total transient time of somewhat less than 100 ns.
The area overhead of the dynamic-biasing approach mainly consists of the generation of the bias voltages, the voltage switches, and the distribution network for the bias voltages. To compare DBB with power gating, the example of Slide 8.33 is revisited. The body-bias circuitry consists of two main blocks: a central bias generator (CBG) and many distributed local bias generators (LBGs). The function of the CBG is to generate a process-, voltage-, and temperature-invariant reference voltage, which is then routed to the LBGs. The CBG uses a scaled-bandgap circuit to generate a reference voltage, which is 450 mV below the main supply — this represents the amount of forward bias to apply in active mode. This reference voltage is then routed to all the distributed LBGs. The function of the LBG is to refer the offset voltage to the supply voltages of the local block. This ensures that any variations in the local supplies will be tracked by the body voltage, maintaining a constant 450 mV of FBB.

To ensure that the impedance presented to the well is low enough, the forward biasing of the ALU required 30 LBGs. Observe that in this study only the PMOS transistors are dynamically biased, and that only forward biasing is used (in standby, zero bias is used). The total area overhead of all the bias units and the wiring turned out to be approximately 8%.

The effectiveness of the DBB approach is demonstrated with an example of an application-specific processor, the SH-mobile from Renesas (also called the SuperH Mobile Application Processor). The internal core of the processor operates at 1.8 V (for a 250 nm CMOS technology). In standby mode, reverse body-biasing is applied to the PMOS (3.3 V) and the NMOS (−1.5 V) transistors. The 3.3 V supply is already externally available for the
I/O pins, whereas the $-1.5$ V supply is generated on-chip. Similar to the power-gating approach, special “switch cells” are included in every row of standard cells, providing the circuitry to modulate the well voltages.

For this particular design, the DBB approach reduces the leakage by a factor of 28 for a fairly small overhead. Unfortunately, what works for 250 nm does not necessarily translate into similar savings in the future.

Slide 8.39
As we had already observed in Slide 2.12, the effectiveness of back biasing reduces with technology scaling. Although for a 90 nm technology, a combination of FBB and RBB may still yield a 150 mV threshold change, the effect is substantially smaller for 65 nm. This trend is not expected to change course substantially in the coming technology generations. The potential savior is the adoption of dual-gate devices, which may be adopted in the 45 nm (and beyond) technology generations. Hence, as for now, DBB is a useful technology up to 90 nm, but its future truly depends upon device and manufacturing innovations.

Slide 8.40
Ultimately, the best way to reduce leakage in standby mode is to ramp the supply voltage all the way down to 0 V. This is the only way to guarantee the total elimination of leakage. A controllable voltage regulator is the preferred way of accomplishing this Supply Voltage Ramping (SVR) scheme. With voltage islands and dynamic voltage scaling becoming common practice (see Chapter 10), voltage regulators and converters are being integrated into the SoC design process, and the overhead of SVR is negligible. In designs where this is not the case, switches can be used to swap the “virtual” supply rail between $V_{DD}$ and GND. As the switches themselves leak, this approach is not as efficient as the ramping.
The overhead of the SVR scheme is that upon reactivation all the supply capacitance has to be charged up anew, leading to a longer start-up time. Obviously all state data are lost in this regime. If state retention is a concern, techniques discussed earlier in the chapter such as the transfer of essential state information to persistent memory or keeping the supply voltage of the state memory above the retention voltage (DRV), are equally valid.

**Slide 8.41**
This slide shows a pictorial perspective of the supply ramping approach (both for ramping down to GND or to the data retention voltage DRV). SVR in concert with dynamic voltage scaling (DVS – see Chapter 10) is at the core of the “voltage island” concept, in which a chip is divided into a number of voltage domains that can change their values dynamically and independently. To have the maximum effect, it is important that signals crossing the boundaries of voltage islands are passed through adequate converters and isolators. This is in a sense similar to the boundary conditions that exist for signals crossing clock domains.

**Slide 8.42**
The impact of SVR is quite important. Because of the exponential nature of the DIBL effect, just reducing the supply voltage from 1 V to 0.5 V already reduces the static leakage power by a factor of 8.5 for a four-input NAND gate in a 90 nm CMOS technology. With the proper precautions (as we will discuss in Chapter 9), data retention may be ensured all the way to 300 mV. However, nothing beats scaling down all the way to ground.
Integration in Standard-Cell Layout Methodology

- Power switch cell easily incorporated into standard design flow
  - Cell has same pitch as existing components
  - No changes required to cell library from foundry
- Switch design can be independent of block size

If voltage ramping is not an option, switching between different rails (from $V_{DDH}$ to GND or $V_{DDL}$) is still a viable alternative, even though it comes with a larger leakage current in standby (through the $V_{DDH}$ switch). The switch to the lower rail ($V_{DDL}$ or GND) can be made small as it only carries a very small amount of current. The SVR approach can be incorporated in the standard design flows in a similar way as the power-gating and the dynamic body-biasing approaches discussed earlier. The only requirement is a number of extra cells in the library with appropriate sizing for the different current loads.

**Standby Leakage Management – A Comparison**

<table>
<thead>
<tr>
<th></th>
<th>Transistor Stacking</th>
<th>Power Gating</th>
<th>Dynamic Body Biasing</th>
<th>Supply Voltage Ramping</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pros</strong></td>
<td>Conventional technology</td>
<td>Conventional technology</td>
<td>Re-use of standard designs</td>
<td>Most effective</td>
</tr>
<tr>
<td></td>
<td>No performance impact</td>
<td>Conceptually simple</td>
<td>No performance impact</td>
<td>Also available in switched version</td>
</tr>
<tr>
<td></td>
<td>Most effective</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Cons</strong></td>
<td>Limited impact</td>
<td>Performance impact of serial transistor</td>
<td>Triple well</td>
<td>Needs voltage regulator or extra rails</td>
</tr>
<tr>
<td></td>
<td>Special registers</td>
<td>Changes in design flow</td>
<td>Slow activation</td>
<td>Slow activation</td>
</tr>
<tr>
<td><strong>Potential Savings</strong></td>
<td>5–10</td>
<td>2–40</td>
<td>2–1000</td>
<td>Huge</td>
</tr>
</tbody>
</table>

That a module is placed in the appropriate standby mode when needed, taking into account the potential savings and the overhead involved. This requires a system-level perspective, and a chip architecture with power management integrated into its core.

**Slide 8.44**

Though standby power reduction is a major challenge, it is also clear that a number of techniques have emerged addressing the problem quite effectively as shown in this overview slide. The main trade-offs are between standby power, invocation overhead, area cost, and runtime performance impact.

The main challenge the designer faces is to ensure that a module is placed in the appropriate standby mode when needed, taking into account the potential savings and the overhead involved. This requires a system-level perspective, and a chip architecture with power management integrated into its core.

**Slide 8.45**

In the end, what is really needed to deal with the standby power problem is an ideal switch, which conducts very small current when off and has a very low resistance when on. Given the importance of standby power, spending either area or manufacturing cost toward such a device seems to be worthwhile investment. In Chapter 2, we discussed some emerging devices that promise steeper sub-threshold slopes, such as, for instance, the dual-gate device. Some speculative transistors even promise slopes lower than 60 mV/dec.
Yet, a switch that can be fully turned off in standby mode would be the ultimate. This is why some of the current research into microelectromechanical systems (MEMS)-based switches is so appealing. A number of research groups are investigating the idea of a MOS transistor with a “movable” gate, where the thickness of the gate insulator is modified using electrostatic forces. This may lead to a switch with ignorable leakage in off mode, yet a good conductivity in the on mode. We believe that devices like this may ultimately play an important role in extending the life of CMOS design into the nanometer scale. Yet, as always, any modification in the manufacturing process comes at a considerable cost.

### Summary and Perspectives

- Today’s designs are not leaky enough to be truly power-performance optimal! Yet, when not switching, circuits should not leak!
- Clock gating effectively eliminates dynamic power in standby
- Effective standby power management techniques are essential in sub-100 nm design
  - Power gating the most popular and effective technique
  - Can be supplemented with body biasing and transistor stacking
  - Voltage ramping probably the most effective technique in the long range (if gate leakage becomes a bigger factor)
- Emergence of “voltage or power” domains

In the next chapter, we shall discuss how the inverse is true for memories, where controlling standby leakage while retaining storage has evolved into one of the most challenging problems for today and tomorrow.
References

Books and Book Chapters


Articles

- T. Kuroda et al., "A 0.9 V 150 MHz 10 mW 4 mm² 2-D discrete cosine transform core processor with variable-threshold-voltage scheme," JSSC, 31(11), pp. 1770–1779, Nov. 1996.

References (cont.)


Slides 8.47 and 8.48

Some references.