The computational load and hence the activity of a processor or an SoC may change substantially over time. This has some profound repercussions on the design strategy for low power, as this means that the optimal design point changes dynamically. The standby case, discussed in the previous chapters, is just a special case of these dynamic variations (with the activity dropping to zero). The concept of runtime optimization in the energy–delay space presents a fundamental departure from traditional design methodology, in which all design parameters such as transistor sizes and supply and threshold voltages were set by the designer or the technology, and remained fixed for the lifetime of the product. Though runtime optimization creates some unique opportunities, it also presents some novel challenges.
Motivation behind runtime optimization
- Dynamic voltage and frequency scaling
- Adaptive body biasing
- General self-adaptation
- Aggressive deployment
- Power domains and power management

Chapter Outline

Slide 10.2
The Chapter starts by motivating the need for dynamic adaptation. A number of different strategies to exploit the variation in activity or operation mode of a design are then described in detail. Dynamic voltage- and body-bias scaling are the best-known examples. Increasingly, it becomes necessary to dynamically adjust a broad range of design parameters, leading to a self-adapting approach. In the extreme case, one can even adjust the design outside the safe operation zone to further save energy. This approach is called “aggressive deployment” or “better than worst-case” design. Finally, we discuss how the adoption of these runtime techniques leads to the need for a power management system, or, in other words, “a chip operating system”.

Power dissipation strong function of activity
In many applications, activity varies strongly over time:
- Example 1: Operational load varies dramatically in general-purpose computing. Some computations also require faster response than others.
- Example 2: The amount of computation to be performed in many signal processing and communication functions (such as compression or filtering) is a function of the input data stream and its properties.

Optimum operation point in the performance–energy space hence varies over time
Changes in manufacturing, environmental, or aging conditions also lead to variable operation points

Designs for a single, fixed operation point are sub-optimal

Slide 10.3
Activity variations and their impact on power are an important reason why runtime optimization had become an attractive idea in the late 1990s. Since then, other important sources of dynamic variations have emerged. Device parameters change over time owing to aging or stress effects, or owing to varying environmental conditions (e.g., temperature). Changes in current loads cause the supply rails to bounce up and down. These added effects have made runtime optimization over the available design parameters even more attractive. Sticking to a single operational point is just too ineffective.

Slide 10.4
To illustrate just how much workloads can vary over time, let us consider the case of a video compression module. A fundamental building block of virtually every compression algorithm is the motion compensation block, which computes how much a given video frame differs from the previous one and how it has changed. Motion compensation is one of the most computationally intensive functions in video compression algorithms such as MPEG-4 and H.264.
One can intuitively understand that the motion compensation module has to work a lot harder in a fast-moving car chase scene than in a slow pan of a nature landscape. This is clearly illustrated in the chart of the lower-right corner, which plots a histogram of the number of IDCTs (inverse discrete cosine transforms) that have to be performed per frame. The distribution is strongly bi-modal. It also shows that the computational effort per frame can vary over 2–3 orders of magnitude.

Variable Workloads in General-Purpose Computing

The same broad distribution holds for general-purpose computing as well. Just watch the “CPU Usage” chart of your laptop for a while. Most of the time, the processor runs at about 2–4% utilization, with occasional computational bursts extending all the way to 100% utilization. Identical scenarios can be observed for other computer classes, such as desktops, workstations, file servers, and data centers. When observing these utilization traces, it becomes quite obvious that there must be an opportunity to exploit the periods of low activity to reduce energy dissipation.

Slide 10.6

As stated in earlier chapters, the variation in activity moves or changes the optimal E–D curve. In addition, the delay expectation may change as well depending upon operating modes. The optimal operation point hence moves, which means that an energy-efficient design should adapt itself to the changing conditions. Unfortunately, the number of knobs that are available to the designer of the runtime system is restricted. Of the traditional design parameters, only supply and threshold voltages are truly available. Dynamically changing the transistor sizes is not very practical or
Adapting to Variable Workloads

- Goal: Position design in optimal operational point, given required throughput
- Useful dynamic design parameters: $V_{DD}$ and $V_{TH}$
  - Dynamically changing transistor sizes non-trivial
- Variable supply voltage most effective for dynamic power reduction

Adjusting Only the Clock Frequency

- Often used in portable processors
- Only reduces power – leaves energy per operation unchanged
  - Does not save battery life

![Clock Frequency Reduction Diagram]

Slide 10.7

Consider, for instance, the case of the microprocessor embedded in a laptop computer. Assume that the computational tasks can be divided into high-performance tasks with short latency requirements, and background tasks, where the latency is not that important. A processor that runs at a fixed frequency and voltage executes both types of tasks in the same way – this means that the high-performance task is executed within specifications (as shown by the dotted lines), whereas the low-end task is performed way too fast. Executing the latter slower would still meet specifications, and offers the opportunity for power and energy savings.

One approach that was adopted by the mobile-computing industry early on is to turn down the clock frequency when the computer is operating on battery power. Lowering the clock frequency reduces the power dissipation ($P = CV^2f$) proportional to the frequency reduction (assuming that leakage is not a factor). However, it comes with two disadvantages:

- The reduced-frequency processor does fine with the high-latency tasks, but fails to meet the specifications for the high-performance functions;
- Though it scales power, this approach does not change the energy per operation. Hence, the amount of work that can be performed on a single battery charge remains the same.
The first concern can be addressed by changing the frequency dynamically in response to the presented workload. *Dynamic Frequency Scaling* (or DFS) makes sure that performance requirements are always met (“just in time computing”), but misses out on the energy reduction opportunity.

**Dynamic Voltage Scaling (DVS)**

- Matches execution speed to requirements
- Minimizes average energy per operation
- Extends battery life by up to one order of magnitude with the exactly same hardware!

A more effective way of exploiting the reduced workload is to scale clock frequency and supply voltage simultaneously (called *dynamic voltage scaling*, or DVS). The latter is possible as frequency scaling allows for higher delays, and hence reduced supply voltages. Whereas pure frequency scaling does reduce power linearly, the additional voltage scaling adds a quadratic factor, and reduces not only the average power but also the energy per operation, while meeting all the performance needs.

![Slide 10.8](image)

**Flashback: \( V_{DD} \) and Throughput**

\[
 f = \left( \frac{V - V_{TH}}{1 - V_{TH}} \right)^\alpha \left( \frac{1}{V} \right)
\]

Where \( f \) and \( V \) are the throughput and supply voltage normalized to the nominal values, and \( \alpha \) is the ratio between threshold and nominal supply voltages.

For \( \alpha = 2 \) and \( V_{DD} \gg V_{TH} \), \( f = \frac{1}{V} \) (long-channel device)

dependency between frequency and voltage is observed. In short-channel transistors, the supply voltage initially scales super-linearly, but the effect saturates for larger reductions in clock frequency.

![Slide 10.9](image)
The results of the previous slide can now be used to compute the energy savings resulting from simultaneous supply and frequency scaling. The resulting chart clearly demonstrates that DVS reduces energy per operation super-linearly. Reductions of the clock frequency by factors of 2 and 4 translate into energy savings by factors of 3.8 and 7.4, respectively (in a 90 nm CMOS technology). Scaling only the frequency would have left the energy per operation unchanged – and taking leakage into account, it might even go up!

The dynamic voltage scaling approach, though very attractive, comes with one major setback: it requires a supply voltage that can be adjusted continuously!

Having to adjust the supply voltage adaptively and continuously may or may not present a substantial overhead depending upon the system the processor is embedded in. Most microprocessor mother boards include sophisticated voltage regulators that allow for a range of
programmable output voltages. However, in other systems the cost of such a regulator might be just too high. Fortunately, the benefits of DVS can also be obtained when only a few discrete supply voltages are available. By dithering the module between the different voltages (i.e., flipping between them on a periodic basis), continuous voltage scaling can be emulated. The resulting energy dissipation per operation now lies on the linear interpolation between the different discrete-voltage operation points. The percentage of time spent at each voltage determines the exact operation point. Adding more discrete supply voltages allows for a closer approximation of the continuous DVS curve. This approach is often called voltage hopping or voltage dithering.

For example, if only one extra supply (at \( V_{\text{DD}/2} \)) is available in addition to the nominal supply, spending equal time at both supplies reduces the energy by a factor of 1.6 (instead of the factor 2 that would have resulted from continuous scaling).

### Challenge: Estimating the Workload

- Adjusting supply voltage is not instantaneous and may take multiple clock cycles
- Efficiency of DVS a strong function of accuracy in workload estimation
- Depending upon type of workload(s), their predictability, and dynamism
  - Stream-based computation
  - General-purpose multi-processing
Example 1: Stream-Based Processing

- Examples: voice or multimedia processing

![Diagram of Stream-Based Processing]

- FIFO measures workload
- Control dynamically adjusts $V_{DD}$ (and hence $f_{clk}$)

[Ref: L. Nielsen, TVLSI'94]

Stream processing is a particular class of applications where the workload estimation is relatively straightforward. The video-compression example of Slide 10.4 belongs to this class, and so do audio and voice compression, synthesis, and recognition. In stream processing, new samples are presented at a periodic rate. When buffering the incoming samples into a FIFO, the utilization of the FIFO is a direct measure of the presented workload.

When it is close to full, the processor should speed up; when it empties, the processor can slow down. An output FIFO then translates the variable processing rate into the periodic signal that may be required by the playback device or the communication channel.

Buffer utilization is only one measure of the workload. Its disadvantage is that it comes with extra latency. More sophisticated estimators can be envisioned. For instance, many signal processing and communication algorithms allow for the construction of simple and quick estimators of the computational effort needed. The outcome of these can then be used to control the voltage-frequency loop.

Slide 10.15

The effectiveness of this approach is shown for the case of an MPEG-4 encoder. Each time a new frame arrives, the “scheduler” estimates the amount of work to be performed before the next milestone, and the voltage is adjusted accordingly. In this particular example, the designers choose to use voltage dithering. Analysis showed that just two discrete voltages were sufficient to reap most of the benefits. A power reduction by a factor of 10 was obtained.
Another challenge of the DVS approach is how to translate a given “request for performance” into voltage, and subsequently into frequency. One option is to use a self-timed approach (as was adopted in the first ever published DVS paper by Nielsen in 1994). This effectively eliminates the voltage-to-frequency translation step. The performance-to-voltage translation is performed by the closed control loop. Important design choices still need to be made: what voltage steps to apply in response to performance requests, and how fast to respond.

In the synchronous approach, the voltage-to-frequency translation can also be achieved dynamically using a closed-loop approach as well. A dummy delay line, mimicking the worst-case critical path, translates voltage into delay (and frequency).

A third approach is to model the voltage–frequency relationship as a set of equations, or as a set of empirical parameters stored in a look-up table. The latter can be obtained by simulation or from measurements when the chip is started up. To account for the impact of variations caused by temperature changes or device aging, the measurements can be repeated on a periodic base. Table look-up can also be used to translate computational requirements into frequency needs.

The closed-loop approach to set the voltage and the frequency is illustrated in this Figure. The difference between the desired and actual clock frequencies is translated into a control signal for the DC–DC converter (after being filtered to avoid rapid fluctuations). The voltage is translated into a clock frequency by the VCO, which includes a replica of the critical path to ensure that all the timing constraints in the processor or computational module are met.
Anyone familiar with phase-locked loops (PLLs) recognizes this scheme. It is indeed very similar to the PLLs commonly used in today’s microprocessors and SoCs to set the clock phase and frequency. The only difference here is that the loop sets the supply voltage as well.

### Table Look-up Frequency–Voltage Translation

<table>
<thead>
<tr>
<th>User Logic</th>
<th>Calibration Unit (Delay analysis)</th>
<th>Frequency-to-Voltage Translation Table (F–V Table)</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ref Clk</td>
<td>vDD</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Voltage–Frequency (V–F) relationship measured at start-up time (or periodically)
- Delay measurements for different voltages obtained from actual module
- Using array of ring oscillators
- Inverse function (F–V) stored in look-up table, taking into account logic structure
- Can compensate for temperature variations

[Ref: H. Okano, VLSI'06]

Although the replica critical path approach is an effective and simple way of relating supply voltage and clock frequency, it faces some important challenges in the deep-submicron age. With process variations causing different timing behavior at different locations on the chip, a single replica circuit may not be representative of what transpires on the die. One option is to combine the results of many distributed replica circuits, but this rapidly becomes complex.

An alternative approach is to calibrate the design at start-up time and record the voltage–frequency relationships (compared to an accurate reference clock) in a table. (It is also possible to create the table in advance using simulations.) One possible implementation of this calibration procedure is to apply a sequence of voltages to the logic module and measure the resulting delay. The inverse function, stored in a look-up table, can then be used to translate the requested frequency into the corresponding supply voltage. This approach can accommodate the impact of temperature changes, as the delay–temperature relationship is known or can be determined by simulation in advance. This information can be used to recalibrate the look-up table. A conceptual diagram of a table-based system is shown in this slide.

Another calibration option is to use an array of ring oscillators of different sizes, effectively measuring the actual process parameters. This process information can then be translated into a voltage using a $P-V$ (process–voltage) table, which is obtained in advanced using simulation. This approach, which was proposed in [Okano, VLSI06], has the advantage that it fully orthogonalizes design- and process-dependent factors.

### Slide 10.18

All the above considerations are equally valid for general-purpose processors. The major difference is that the workload-to-voltage (or frequency) translation is typically performed in software. Actually, it becomes an operating-system function, the task of which is to translate a set of computational deadlines into a schedule that meets the timing requirements. The processor frequency is just one of the extra knobs that can be added to the set the OS has at hand.

A simple way of estimating the desired clock frequency is to divide the expected number of cycles needed for the completion of the task(s) by the allotted time. For the example of the MPEG
encoder, for instance, the desired clock frequency can be obtained empirically or through simulation. Multi-tasking and the intricacies of modern microprocessors make this translation a lot more complex, however, and more complex workload and frequency estimates are needed.

Oracle: perfect knowledge of the future
Zero: heuristic scheduling algorithm
Largest savings for less-demanding or bursty apps (UI, audio)
Difficult to get large gains in computation-intensive code (MPEG)

The effectiveness of DVS in general-purpose processing is by no small means determined by the quality of the voltage-scheduling algorithm. Task scheduling has been studied intensively in a number of fields such as operations research, real-time operating systems, and high-level synthesis. Much can be learned from what has been developed in those fields.

The maximum savings would be obtained by a so-called Oracle scheduler, which has perfect foreknowledge of the future (as well as all the costs and overheads incurred by a change in voltage), and hence can determine the perfect voltage for every task at hand. The quality of any other scheduling algorithm can be measured by how close it gets to the Oracle schedule.

The worst-performing scheduler is the “ASAP” approach. This puts the processor in idle mode (and associated voltage) when there is no activity, and ramps to the maximum voltage whenever a computation has to be performed.

Most practical scheduling algorithms rely on a number of heuristics to determine the order of task execution as well as to select the design parameters. An example of such is the “zero” algorithm that was proposed in [Pering99].

From the results presented in the table, a number of interesting observations can be drawn.

Slide 10.20
The effectiveness of DVS in general-purpose processing is by no small means determined by the quality of the voltage-scheduling algorithm. Task scheduling has been studied intensively in a number of fields such as operations research, real-time operating systems, and high-level synthesis. Much can be learned from what has been developed in those fields.

The maximum savings would be obtained by a so-called Oracle scheduler, which has perfect foreknowledge of the future (as well as all the costs and overheads incurred by a change in voltage), and hence can determine the perfect voltage for every task at hand. The quality of any other scheduling algorithm can be measured by how close it gets to the Oracle schedule.

The worst-performing scheduler is the “ASAP” approach. This puts the processor in idle mode (and associated voltage) when there is no activity, and ramps to the maximum voltage whenever a computation has to be performed.

Most practical scheduling algorithms rely on a number of heuristics to determine the order of task execution as well as to select the design parameters. An example of such is the “zero” algorithm that was proposed in [Pering99].

From the results presented in the table, a number of interesting observations can be drawn.
The savings that can be obtained by DVS and voltage scheduling are strongly dependent upon the applications at hand. The largest savings occur for applications that do not stress the processor, such as user-interface interactions or audio processing. On the other hand, only small gains are made for demanding applications such as video (de)compression. This is why it is worthwhile to farm these applications out to co-processors, as discussed in Chapter 5.

Investing in a good scheduling algorithm is definitely worthwhile. Getting within a few percentiles from the Oracle scheduler is possible.

Slide 10.21
The impact of using different scheduling algorithms is illustrated in this slide for a "bursty" user-interface application. The supply voltage (and hence energy) levels as obtained by the "ASAP" and "zero" algorithms are compared. The latter raises the supply voltage only rarely above the minimum, does so only for latency-sensitive tasks, and never needs the maximum voltage.

Slide 10.22
The voltage- and frequency-setting loop for general-purpose processors pretty much follows the scheme detailed in Slide 10.17. The actual clock frequency $F_{CLK}$ is translated into a digital number by a counter–latch combination (sampled at a frequency of 1 MHz, in this example). The result is compared with the desired frequency, as set by the operating system. The goal of the feedback loop is to get the error frequency $F_{ERR}$ as close as possible to zero.

After filtering, the error signal is used to drive the DC–DC converter (which is an inductive buck converter in this particular case). The supply voltage is translated into the clock frequency $F_{CLK}$ with the aid of a ring oscillator, which matches the critical path of the processor [Burd’00].
A High-Performance Processor at Low Energy

If processor in low-performance mode most of the time:
85 MIPS processor @ 0.54 mW/MIPS

[Ref: T. Burd, JSSC'00]

0.54 μJ/instruction, or anything in between. If the duty cycle is low, which is often the case in embedded application processors, DVS creates a high-performance processor with a very low average energy per operation. The operational point just moves back and forth on the energy–delay (performance) curve.

Examples of DVS-Enabled Microprocessors

- Early Research Prototypes
  - Toshiba MIPS 3900: 1.3–1.9 V, 10–40 MHz [Kuroda98]
  - Berkeley ARM8: 1.2–3.8 V, 6–85 MIPS, 0.54–5.6 mW/MIPS [Burd00]
- Xscale: 180 nm 1.8 V bulk-CMOS
  - 0.7–1.75 V, 200–1000 MHz, 55–1500 mW (typ)
  - Max. Energy Efficiency: ~23 MIPS/mW
- PowerPC: 180 nm 1.8 V bulk-CMOS
  - 0.9–1.95 V, 11–380 MHz, 53–500 mW (typ)
  - Max. Energy Efficiency: ~11 MIPS/mW
- Crusoe: 130 nm 1.5 V bulk-CMOS
  - 0.8–1.3 V, 300–1000 MHz, 0.85–7.5 W (peak)
- Pentium M: 130 nm 1.5 V bulk-CMOS
  - 0.95–1.5 V, 600–1600 MHz, 4.2–31 W (peak)
- Extended to embedded processors (ARM, Freescale, TI, Fujitsu, NEC, etc.)

Slide 10.23
Instead of representing a single operational point, a DVS processor moves dynamically in the energy–delay space. This is illustrated quite nicely with the graph shown in this slide, which plots the operational performance versus the energy, for one of the first DVS processors, published at ISSCC in 2000. Implemented in a 600 nm technology(!), the same processor can implement either an 85 MIPS ARM processor operating at 6.5 μJ/instruction or a 6 MIPS processor at Dynamic V<sub>DD</sub>.

Slide 10.24
DVS has progressed immensely since its introduction by the research community. Today, a wide range of embedded, DSP, and notebook processors have embraced the concept. In typical applications, the energy per instruction can vary by as much as a factor of 10.

Slide 10.25
Although dynamic voltage scaling seems to be a no-brainer for a number of applications (if of course, continuous or discrete supplies can readily be made available), there existed a lot of resistance against its adoption in the early days. The main concerns were related to how it would be possible to guarantee that timing conditions and signal integrity are met under changing conditions. It is already a major challenge to verify that a processor functions correctly for a single
supply voltage – a task that is getting more complicated with the increasing influence of process variations. Imagine now what it takes if the supply voltage is varied dynamically. Must one check correct functionality at every supply voltage within the operation range? What about the transient conditions while the voltage is being ramped? Must one halt the processor during that time, or can it keep on running?

All these questions are very pertinent. Fortunately, a number of key properties make the verification task a lot simpler than what would be expected.

**DVS Challenge: Verification**

- Functional verification
  - Circuit design constraints
- Timing verification
  - Circuit delay variation
- Power distribution integrity
  - Noise margin reduction
  - Delay sensitivities (local power grid)

**Need to verify at every voltage operation point?**

Design for Dynamically Varying $V_{DD}$

- Logic needs to be functional under varying $V_{DD}$
  - Careful choice of logic styles is important (static versus dynamic, tri-state busses, memory cells, sense amplifiers)
- Also: need to determine max $|dV_{DD}/dt|$
Let us consider the case of complementary CMOS logic. A positive property of this most popular logic style is that the output is always connected to either GND or $V_{DD}$ through a resistive path (during a transition it may temporarily be connected to both). If the output is high and the supply voltage changes, the output of the gate just tracks that change with a short delay owing to the $RC$ time constant. Hence, the functionality of the logic is by no means impacted by DVS. The same is true for a static SRAM cell. In fact, static circuits continue to operate reliably even while the supply voltage is changing.

This is not the case for dynamic circuits. During evaluation, the “storage” node of the circuit may be at high impedance, and disconnected from the supply network. Ramping the supply voltage during that time period can lead to a couple of failure modes:

- When the supply voltage rises during evaluation, the “high” signal on the storage node drops below the new supply voltage. If the change is large enough ($> V_{TH,PMOS}$), it may be considered a logic-low by the connecting gate.
- On the other hand, when the supply voltage is ramped down, the stored node voltage rises above the supply voltage, and may cause the onset of latch-up, if the difference is larger than the $V_{be}$ of the parasitic bipolar transistor.

These failure mechanisms can be avoided by either keeping the supply voltage constant during evaluation, or by ramping the rails slowly enough that the bounds, defined above, are not exceeded.

High-impedance tri-state busses should be avoided for the same reason.
DVS System Transient Response

Ring oscillator (for \( \frac{dV_{DD}}{dt} = 20 \text{ V/\mu s} \))

Output \( f_{clk} \) instantaneously adapts to new \( V_{DD} \)

[Ref: T. Burd, JSSC’00]

Relative Timing Variation

Delay relative to ring oscillator

Four extreme cases of critical paths:

- Gate
- Ring oscillator
- Interconnect
- Diffusion
- Series

Delay for all components varies monotonically with \( V_{DD} \)
Timing verification only needed at min & max \( V_{DD} \)

[Ref: T. Burd, UCB’01]

Slide 10.29
The simulated response of a CMOS ring oscillator, shown in this slide, amply serves to validate our argument that static CMOS keeps performing correctly while the voltage is ramped. The plot shows how the output clock signal \( f_{clk} \) keeps on rising while the supply voltage increases.

Slide 10.30
Even if a circuit works correctly at one voltage from a timing perspective, this by no means guarantees that it also does so at another one. The relative delays from modules in different logic styles may change owing to the voltage scaling. If so, it may be necessary to check the timing at every supply voltage in the operation range.

To evaluate what transpires during voltage scaling, the relative delay (normalized to the delay of a ring oscillator) versus supply voltage is plotted for four typical circuit elements. These include inverter chains, of which the loads are dominated by gate, interconnect, and diffusion capacitance (as each of these has a different voltage dependence). To model paths dominated by stacked devices, a fourth chain consisting of 4 PMOS and 4 NMOS transistors in series is analyzed as well. The relative delay of all four circuits is at a maximum at only the lowest or highest operating voltages, and is either monotonically falling or rising in between. This means that it is sufficient to ensure timing compliance at the extreme ends of the supply voltage range to guarantee compliance everywhere in between. This substantially reduces the timing verification effort.
Note: it may be possible to create a relative-delay curve with a minimum or a maximum occurring in-between the end points, by combining circuits of the different types. However, because the gate-capacitance-dominated delay curve is convex, whereas the others are concave, the combination typically results in a rather flat curve, and the observation above pretty much still holds.

\[
\frac{\partial \text{Delay}}{\partial \text{Delay}} = \frac{\partial \text{Delay}}{\partial V_{DD}} \cdot \frac{\Delta V_{DD}}{\text{Delay}(V_{DD})}, \quad \text{where } \Delta V_{DD} = I(V_{th}) \cdot R
\]

Another concern is the effect of supply bounce as it may induce timing variations and potential violations. We are not concerned about global supply voltage changes as they affect all timing paths equally and the clock period is adjusted as well – remember that the clock frequency is derived from the supply voltage in a DVS system.

Localized supply variations, however, may only affect the critical paths, and not the clock generator, and can lead to timing violations if the local supply drop is sufficiently large. As such, careful attention has to be paid to the local supply routing. As always, a certain percentage of the timing budget must be set aside to accommodate the impact of supply bounce. However, the question again arises as to the voltage at which the impact of supply noise is the largest and whether we should check it for the complete range.

The sensitivity of delay with respect to $V_{DD}$ can be quantified analytically, and the normalized result is plotted as a function of $V_{DD}$ in this slide. For a submicron CMOS process, the delay sensitivity peaks at approximately $2V_{TH}$. Thus, in the design of the local power grid, we only need to ensure that the resistive (inductive) voltage drop of the power distribution grid meets the design margins for one single supply voltage (i.e., $2V_{TH}$). This is sufficient to guarantee that they are also met at all other voltages.

All in all, though the DVS approach undoubtedly increases the verification task, the extra effort is bounded. In fact, one may even argue that the adaptive closed loop actually simplifies the task somewhat as some process variations are automatically adjusted for.

Slide 10.32
So far, we have only considered the dynamic adaptation of the supply voltage. In line with our discussions on design-time optimization, it seems only natural to consider adjusting the threshold voltages at runtime as well. This approach, called Adaptive Body Biasing or ABB, is especially appealing in light of the increasing impact of static power dissipation. Raising the thresholds when
Adapative Body Biasing (ABB)

- Similar to DVS, transistor thresholds can be varied dynamically during operation using body biasing
- Extension of DBB approach considered for standby leakage management
- Motivation:
  - Extends dynamic E–D optimization scope (as a function of activity)
  - Helps to manipulate and control leakage
  - Helps to manage process and environmental variability (especially $V_{TH}$ variations)
  - Is becoming especially important for low $V_{DD}/V_{TH}$ ratios

The activity is low (and the clock period high), and lowering them when the activity is high and the clock period short, seems to be a perfect alternative or complement to the DVS approach. It should be apparent that ABB is the runtime equivalent of the Dynamic Body Biasing (DBB) approach, introduced in Chapter 8 to address standby leakage.

In addition to dynamically adjusting the static power, ABB can help to compensate for some of the effects introduced by static or dynamic threshold variations – caused by manufacturing imperfections, temperature variations, aging effects, or all of the above. In fact, if well-executed, threshold variations can be all but eliminated.

Slide 10.33
As can be observed, variations in thresholds may cause the performance of a module to vary substantially. This effect is more pronounced when the supply voltage is scaled down and the $V_{DD}/V_{TH}$ ratio reduced. Though a 50 mV change in threshold causes a delay change of only 13% at a supply voltage of 1 V (for a 90 nm CMOS technology), it results in a 55% change when the supply is reduced to 0.45 V.

Slide 10.34
The idea of using ABB to address process variations was already introduced in 1994 [Kobayashi94] in a scheme called SATS (self-adjusting threshold voltage scheme). An on-chip leakage sensor amplifies the leakage current (the resistive divider biases the NMOS transistor for maximum gain). When the leakage current exceeds a preset threshold, the well bias generation circuit is turned on, and the reverse bias is increased by lowering the well voltage. The same bias is used for all NMOS
transistors on the chip. Though the circuit shown in the slide addresses the threshold adjustment of the NMOS transistors, the thresholds of the PMOS devices also can be controlled in a similar way.

Note that the overall goal of the SAT scheme is to set leakage to a specific value; that is, the transistor thresholds are set to the lowest possible value that still meets the power specifications.

**Slide 10.35**

The effectiveness of the SATS is quite apparent from the measured results shown in this chart. Even with the raw threshold varying by as much as 300 mV, the control loop keeps the actual threshold within a 50 mV range.

**Slide 10.36**

This slide features a more recent study of the potential of adaptive body biasing. A test chip implemented by a group of researchers at Intel in a 150 nm CMOS technology [Tschanz02] features 21 autonomous body-biasing modules. The idea is to explore how ABB can be exploited to deal not only with inter-die, but also with intra-die variations. Both the reverse and forward body biasing options are available. Each sub-site contains a replica of the critical path of the circuit under test (CUT), a phase detector (PD) comparing the critical path delay with the desired clock period, and a phase-to-bias converter consisting of a counter, a D/A converter, and an op-amp driver. Only PMOS threshold control is implemented in this
The economic impact of applying ABB should not be ignored either. In the microprocessor world, it is common to sort manufactured dies into frequency bins based on the measured performance (in addition, all of the accepted dies should meet both functionality and maximum power requirements). Without ABB, a majority of the dies ends up in the not-so-lucrative low-frequency bin, whereas a large fraction does not meet specifications at all. The application of per-die and (even more) within-die ABB manages to move a large majority to the high-frequency bin, while pushing parametric yield close to 100%.

From this, it becomes apparent that adaptively tuning a design is a powerful tool of the designer in the nanometer era.
ABB is even more effective in circuits operating at low supply voltages and low $V_{DD}/V_{TH}$ ratios. Under those conditions, a small variation in the threshold voltage can cause either a large performance penalty or a major increase in energy consumption. This is illustrated by the distance between the “nominal” and “worst-case” E–D curves of a complex arithmetic logic function, implemented in a 130 nm CMOS technology. The supply voltage is variable and ranges between 200 and 500 mV, whereas the threshold voltage is kept constant. Observe that the delay is plotted on a logarithmic scale.

A substantial improvement is made when we allow the threshold voltages to be tuned. One option is to simultaneously modify all threshold voltages of a modules by adjusting the well voltage. Even then, the worst-case scenario still imposes a performance penalty of a factor of at least two over the nominal case. This difference is virtually eliminated if the granularity of threshold adjustment is reduced—e.g., allowing different body bias values for every logical path. Overall, introducing “adaptive tuning” allows a performance improvement by a factor of 12 over the worst-case un-tuned scenario, while keeping energy constant.

At this point, it is only a small step to consider the advantages of simultaneously applying DVS and ABB. Whereas DVS mainly addresses the dynamic power dissipation, ABB serves to set the passive power to the appropriate level. This combined action should lead to E–D curves that are superior to those obtained by applying the techniques separately.

An example of a circuit incarnation that adjusts both $V_{DD}$ and $V_{TH}$ is shown on this slide [Miyazaki02]. The requested workload is translated into a desired supply
voltage using the table look-up approach. Given the selected $V_{DD}$, a replica of the critical path is used to set the well voltages for NMOS and PMOS transistors so that the requested clock frequency is met. This approach obviously assumes that the replica path shows the same variation behavior as the actual processor.

Slide 10.40
Actual measurements for the circuit of Slide 10.39 indeed show the expected improvements. Compared to DVS only, adding ABB improves the average performance of the circuit substantially for the same power level (and vice versa).

Slide 10.41
Although these performance improvements are quite impressive, one may question how effective the combined DVS/ABB approach is in suppressing the effects of (threshold) variations. The chart on the left side of the slide plots the measured clock frequency and power numbers for the same circuit as collected from a large number of dies (over different wafers). For these measurements, supply voltages were fixed to the nominal value, and no body biasing was applied. Though the measurements show a very broad and wide distribution (20% in both clock frequency and power), a general trend can be observed – that is, slower circuits consume less power (this obviously is not a surprise!).

With the introduction of DVS and ABB, circuits that do not meet the performance or power specification are adjusted and brought within the acceptable bounds (with the exception of some circuits that cannot be corrected within the acceptable range of supply and bias voltages, and hence should be considered faulty). The resulting distribution is plotted on the right, which indicates that
dynamic adaptation and tuning is indeed a very effective means of addressing the impact of technology and device variations.

One very important caveat should be injected here: just when device variations are becoming a crucial design concern, one of the most effective means of combating them – that is body biasing – is losing its effectiveness. As we had already indicated in Chapter 2, the high doping levels used in sub-100 nm technologies reduce the body-effect factor: at 65 nm and below, ABB maybe barely worth the effort (if at all). This is quite unfortunate, and is hopefully only temporary. The introduction of novel devices, such as dual-gate transistors, may restore this controllability at or around the 32 nm technology node.

A Generalized Self-adapting Approach

**Motivation:** Most variations are systematic or slow, and can be measured and adjusted for on a periodic basis

- Parameters to be measured: temperature, delay, leakage
- Parameters to be controlled: \(V_{DD}\), \(V_{TH}\) (or \(V_{BB}\))

- Achieves the maximum power saving under technology limit
- Inherently improves the robustness of design timing
- Minimum design overhead required over the traditional design methodology

---

Slide 10.42

Another important and general observation is worth making. The DVS and ABB schemes, presented in the previous slides, are great examples of a new class of circuits (called self-adaptive) that deal with variations (be it caused by changes in activity, manufacturing, or the environment) by using a closed feedback loop. Online sensors measure a set of indicative parameters such as leakage, delay, temperature, and activity. The resulting information is then used to set the value of design parameters such as the supply voltage and the body bias. In even more advanced schemes, functions might even be moved to other processing elements if performance requirements cannot be met.

The idea is definitely not new. In the 1990s, high-performance processors started to incorporate temperature sensors to detect over-heating conditions and to throttle the clock frequency when the chip got too hot. The difference is that today’s self-adaptive circuits (as adopted in high-end products) are a lot more sophisticated, use a broad range of sensors, and control a wide range of parameters.

---

Slide 10.43

Although adaptive techniques go a long way in dealing with runtime variability, ultimately their effectiveness is limited by the “worst-case” conditions. These may be the voltage at which the timing constraints of a critical path cannot be met or when a memory cell fails. In a traditional-design approach, this is where the voltage scaling ends. However, on closer inspection, one realizes that these worst-case conditions occur only rarely. Hence, if we can cheaply detect the occurrence of such a condition and correct the resulting error when it occurs, we could over-scale the voltage, further reducing the energy dissipation.

Let us, for instance, consider the case of an SRAM memory. As we had discussed in Chapter 9, the minimum operational voltage of an SRAM cell (the DRV) varies cell-by-cell. Fortunately, the
measured distribution of the DRVs over a large memory block shows a long tail. This means that lowering the voltage below the worst case causes some errors, but only a few. The reduction in leakage currents far outweighs the cost of error detection and correction.

Aggressive Deployment (AD)
- Also known as “Better-than-worst-case (BTWC) design”
- Observation:
  - Current designs target worst-case conditions, which are rarely encountered in actual operation
- Remedy:
  - Operate circuits at lower voltage levels than allowed by worst case, and deal with the occasional errors in other ways

Aggressive Deployment – Concepts
- Probability of hitting tail of distribution at any time is small
  - Function of critical-path distribution, input vectors, and process variations
- Worst-case design expensive from energy perspective
  - Supply voltage set to worst case (+ margins)
- Aggressive deployments scales supply voltage below worst-case value
  - “Better-than-worst-case” design strategy
  - Uses error detection and correction techniques to handle rare failures

Hence, knowing the distribution of the critical parameters is important.
- The worst-case scenario leaves a large number of crumbs on the table. All circuitry in a module consumes way too much energy just because of a small number of outliers.
- Hence it pays to let some errors occur by over-scaling, and condone the small overhead of error detection and correction.

Example:
Operate memory at voltages lower than that allowed by worst case, and deal with the occasional errors through error correction

Distribution ensures that error rate is low

Slide 10.44
The basic concepts upon which this “better-than-worst-case” (BTWC) (first coined as such by Todd Austin) design is built are as follows:
- Over-scaling of the supply voltage leads only to rare errors, not to catastrophic breakdown. In the latter case, the overhead of dealing with the errors would dominate the savings.
Like DVS and ABB, BTWC (very often also called aggressive deployment, or AD) relies on the presence of a feedback loop, positioning the system at its optimal operation point from a performance/energy perspective. A BTWC system consists of the following elements:

- A mechanism for setting the supply voltage based on the understanding of the trade-off between introducing errors and correcting them. In one way or another, this control mechanism should be aware of the error distribution (either by simulation in advance, or by adaptive learning).
- An error-detection mechanism – As this function is running continuously, its energy overhead should be small. Coming up with efficient error-detection approaches is the main challenge in the conception of BTWC systems.
- An error-correction strategy – As errors are expected to be rare, it is ok to spend some effort in correcting them. The correction mechanisms can vary substantially, and depend upon the application area as well as the layer in the abstraction chain where the correction is performed.

It is important to realize that the BTWC approach is very generic, and can be applied at many layers of the design abstraction chain (circuit, architecture, system) and for a broad range of application spaces, some of which are briefly discussed in the following slides.
Once we start lowering the supply voltages, some timing paths may not be met and errors start to appear. In the FPGA prototype, the first errors occur at 1.54 V. Observe that the y-scale of this plot is logarithmic. If we would allow for a 1.3% error rate (which means that one out of 75 samples is wrong), the supply voltage can be scaled all the way down to 1.36 V. This translates into a power reduction of 35%.

### Slide 10.46
As a first example, let us consider what happens when we lower the supply voltage of a logical module such as a multiplier, which typically has a wide distribution of timing paths. In a traditional design, the minimum supply voltage is set by the worst-case timing path with an extra voltage margin added for safety. The example on the slide shows the results for an 18x18 multiplier, implemented on an FPGA. Including the safety margin, the minimal operational voltage equals 1.69 V.

### Slide 10.47
It is worth observing that error rate and the shape of the error histogram are functions of the data patterns that are applied. For the example of a multiplier, random data patterns tend to trigger the worst-case paths more often than the correlated data patterns that commonly occur in signal-processing applications. The same holds for many other computational functions, as is illustrated in this slide for a Kogge–Stone adder. When applying data patterns from applications such as bzip or ammp, the voltage can be scaled down by an extra 200 mV for the same error rate.

Hence, to be effective, the voltage-setting module must somehow be aware of the voltage-to-error function. As for DVS, this information can be obtained by simulation or during a training period and can be stored in a table.
Error Detection
- Double-sampling latches (latch + shadow latch) detect timing errors
- Second sample is correct-by-design

Error Correction
- Micro-architectural support restores state
- Timing errors treated like branch mispredictions

Challenges: metastability and short-path constraints

[Ref. D. Ernst, Micro’03]

Based on these observations, it seems worthwhile to reduce the supply voltage below its worst-case value, assuming that a simple mechanism for detecting timing errors can be devised. One way of doing so (called RAZOR) is shown here. Every latch at the end of a critical timing path is replicated by a “shadow latch”, clocked a certain time $\Delta T$ later. The value captured in the main latch is correct, if and only if the shadow latch shows an identical value. If on the other hand, the main clock arrives too early and a path has not stabilized yet, main latch and shadow latch will capture different values. A sole XOR is sufficient to detect the error.

The above description is somewhat over-simplifying, and some other issues need to be addressed for this approach to work. For instance, no shadow latches should be placed on “short paths” as this may cause the shadow latch to catch the next data wave. In other words, checking the set-up and hold-time constraints becomes more complicated. Also, the first latch may get stuck in a metastable state, leading to faulty or undecided error conditions. Extra circuitry can be added to get around this problem. For more detailed information, we refer the interested reader to [D. Ernst, MICRO’03].

Upon detection of an error, a number of strategies can be invoked for correction (depending upon the application space). For instance, since the original RAZOR targets microprocessors, it passes the correction task on to the micro-architectural level.
Slide 10.49
In a sense, an error in the data-path pipeline is similar to a branch misprediction. Upon detection of an error, the pipeline stalled, a bubble can be inserted, or the complete pipeline flushed. One interesting observation is that, upon error, the correct value is available in the shadow register. It can hence be re-injected in the pipeline at the next clock cycle, while stalling the next instruction. To state it simply, a number of techniques are available to the micro-architecture designer to effectively deal with the problem. It comes with some cycle (and energy) overhead, but remember: errors are expected to occur only rarely if the voltage-setting mechanism works correctly.

Slide 10.50
The voltage-setting mechanism is a crucial part of any AD scheme. In the RAZOR scheme, the errors per clock cycle occurring in the data path are tallied and integrated. The error rate is used to set the supply voltage adaptations. As mentioned earlier, knowledge of the voltage-error distribution helps to improve the effectiveness of the control loop.

Slide 10.51
Given our previous discussion of adaptive optimizations in the delay–energy space, it should come as no surprise that BTWC schemes converge to an optimal supply voltage that minimizes the energy per operation. Reducing the supply voltage lowers the energy, but at the same time increases the correction overhead. If the voltage–error relationship is gradual (such as the ones shown for the multiplier and the Kogge–Stone adder), the optimal operational point shows a substantial improvement in energy for a very small performance penalty.
Trade-off curves such as the one shown are typical for any BTWC approach, as will be demonstrated in some of the subsequent slides. **Caveat:** For aggressive deployment schemes to be effective, *it is essential that the voltage–error distribution has a “long tail”*. This means that the onset of errors should be gradual once the supply voltage is dropped below its worst-case value. The scheme obviously does not work if a small reduction leads to “catastrophic failures”. Unfortunately, a broad range of the energy-reduction techniques, introduced earlier in this book, tend to create just this type of condition. Design-time techniques, such as the use of multiple supply and threshold voltages as well as transistor sizing, exploit the slack on the non-critical paths to minimize energy dissipation. The net result of this is that a larger percentage of timing paths become critical. Under such conditions, a small voltage reduction can lead to a catastrophic breakdown. This opens the door for an interesting discussion: wouldn’t it be better to forgo the design-time optimizations and let the runtime optimizations do their job – or vice versa? The only way to get a relevant answer to this question is to exploit the systematic system-level design exploration framework, advocated in this book.

**The Industrial Experience**

- Under typical case conditions all chips are at least 39% more energy-efficient
- Worst-case design uses margins for corners that are very infrequent, or even impossible
- Typical-case operation requires an understanding of when and how systems break!

[Courtesy: K. Flautner, ARM Ltd]

Optimizing Power @ Runtime – Circuits and Systems

Slide 10.52

Although concepts such as runtime adaptation and BTWC design show great potential, it takes substantial effort to transfer them into producible artifacts. Similar to DVS, AD requires a re-evaluation of the standard design flows and a rethinking of traditional design concepts. As we had mentioned in the previous slide, concepts such as RAZOR require an understanding of how and when a chip breaks. To make the approach more effective, we may even want to rethink accepted design technologies. But the benefits of doing so can be very substantial.
In this slide, the impact of applying the RAZOR concept to an embedded processor of the ARM™-family is shown. An energy reduction of at least 39% over all processors is obtained, whereas average savings are at least 50%. Getting to this point required a major redesign not only of the data path but also of the memory modules. But the results show that that effort is ultimately very rewarding.

**Slide 10.53**
The RAZOR concept combines error detection at the circuit level with error correction at the micro-architecture level. Many other BTWC strategies can be envisioned. For example, in this slide we show an approach that employs both error detection and correction at the algorithm level.

One interesting property of many signal-processing and communication applications is that the theory community has provided us with simple ways to estimate the approximate outcome of a complex computation (based on the past input stream). The availability of such estimates provides us with a wonderful opportunity to reduce energy through BTWC.

The “Main Block” in the diagram represents some complex energy-intensive algorithm, such as for instant motion compensation for video compression. In normal operation, we assume this block to be error-free. Assume now that we aggressively scale the supply voltage of this block so that errors start to occur. In parallel with the “Main Block”, a simple estimator is run which computes the expected outcome of the “Main Block”. Whenever the latter ventures to values far from the prediction, an error condition is flagged (detection), upon which the faulty outcome is replaced by the estimation (correction). This obviously deteriorates the quality of the processor – in signal-processing speak, it reduces the signal-to-noise ratio (SNR). However, if the estimator is good enough, the increase in the noise level is masked by the noise of the input signal or by the added noise of the signal-processing algorithm, and hence barely matters. Also observe that “small errors” (errors that only effect the least-significant bits [LSBs]) may go undetected, which is ok as they only impact the SNR in a minor way.

As for RAZOR, algorithmic BTWC leads to an optimal supply voltage. If the error rate gets higher, the error correcting overhead starts to dominate (in addition, the deterioration in SNR may not be acceptable). For this scheme to work, clearly it is essential that the estimator does not make any errors itself. This requires that the “Estimate Module” be run at the nominal voltage. Since it is supposed to be a simple function, its energy overhead is small.
a reduced sampling rate (through sub-sampling). Only the MSAD is voltage scaled. A pleasant surprise is that in the presence of process variations, the AD version performs better than the original one, from an SNR perspective. It turns out that this is not an exception—techniques that exploit the joined statistics of the application and the process often end up performing better than those that don’t.

**Slide 10.55**

As mentioned, the concepts of runtime adaptation and AD are broad and far-reaching. We have only shown a couple of examples in this chapter. A number of other instantiations of the concept are enumerated on this slide. We also suggest that you consult the March 2004 issue of the IEEE Computer Magazine, which features a variety of BTWC technologies.
or at the data retention level, while other modules are active and require either the full supply or a dynamically varying one. Each chip partition that needs individual power control is called a **power domain (PD)**.

The introduction of power domains in the standard design methodology comes with some major challenges. First of all, generating and distributing multiple variable supply voltages with a reasonable efficiency is not trivial. Many of the gains made by varying supply and well voltages could be lost if the voltage conversion, regulation, and distribution is not done efficiently. Another, often forgotten, requirement is that signals crossing power boundaries should be carefully manipulated. Level conversion, though necessary, is not sufficient. For instance, the output signals of an active module should not cause any activity in a connected module in standby; or, vice versa, the grounded output signals of a standby module should not result in any erroneous activity in a connected active block.

The most important challenge however is the **global power management** – that is, deciding what voltages to select for the different partitions, how fast and how often to change supply and well voltages, when to go in standby or sleep mode, etc. In the preceding slides and chapters, we had introduced voltage-setting strategies for individual modules. A distributed approach, in which each module individually chooses its preferred setting at any point in time, can be made to work. Yet, it is often sub-optimal as it lacks awareness of the global state of the system. A centralized **power manager (PM)** often can lead to far more efficient results.

Slide 10.57
There are a couple of reasons for this. First of all, the PM can examine the global state of the system, and may have knowledge of the past state. It is hence in a better position to predict when a block will become active or inactive, or what the level of activity may be. Furthermore, transferring the state to a centralized module allows a sub-module to go entirely dormant, reducing leakage power. For instance, many sleep strategies often employ timers to set the next wake-up time (unless an input event happens earlier). Keeping the timers running eliminates the possibility of complete power-down of the unit. Hence, transferring the time-keeping to a centralized “scheduler” makes clear sense.

Although many SoCs employ some form of a power management strategy, most often it is constructed ad hoc and after the fact. Hence, a methodological approach such as the one advocated
in this slide is advisable. A coordinated PM contains the following components: a central control module (called event/command dispatcher), and time, power, and clock sub-systems. The latter contain the necessary knowledge about past and future timing events, power- and voltage-setting strategies for the individual modules, and the voltage–clock relationships, respectively. The dispatcher uses the information of the three sub-systems to set a voltage-scheduling strategy for the different power domains on the chip. Inputs from PDs (such as a request to shut down, or a request to set up a channel to another PD), as well as scheduling decisions and power-setting commands are interchanged between the PDs and the PM over a “power network” with standardized interfaces.

In a sense, the PM takes on some of the tasks that typically would be assigned to a scheduler or an operating system (OS) (which is why another often-used name for the PM is the “chip OS”). However, the latter normally runs on an embedded processor, and consequently that processor could never go into standby mode. Dedicating the PM functionality to a specialized processor avoids that problem, with the added benefit that its energy-efficiency is higher as well.

**Managing the Timing**

- **Basic scheduling schemes**
  - Reactive
    - Sleep when not actively processing
    - Wake up in response to a pending event
  - Stochastic
    - Sleep if idle and probably not needed in near future [Simunic’02]
    - Wake up on account of expected event in the near future

- **Metrics**
  - Correctness – PD awake when required to be active
  - Latency – time required to change modes
  - Efficiency – minimum total energy consumption [Liao’02]
    - Minimum idle time – time required for savings in lower-power mode to offset energy spent for switching modes

\[
\text{Min. Idle Time} = \frac{E_{\text{loss}}}{P_{\text{awake}}} = \frac{E_{\text{switch}} - P_{\text{idle,switch modes}}}{P_{\text{awake}} - P_{\text{idle}}}
\]

PhD theses of Tajana Simunic (Stanford) and Mike Sheets (UCB) probably present the most in-depth treatments on the topic so far.

**Slide 10.58**

In the literature, a number of PM-scheduling strategies have been proposed. The main metrics to judge the quality of a scheduling strategy are the “correctness” – that is, for instance, having a PD in inactive mode, when another PD attempts to communicate with it might be catastrophic – latency and energy efficiency. The scheduling strategies can be roughly divided into two classes: reactive (based on events at the signaling ports) and proactive. The
Interfacing Between Power Domains

Separate internal logic of block from its interfaces
1. Communicate with other PDs by bundling related signaling into “ports”
   - Communication through a port requires permission (session-based)
   - Permission is obtained through power-control interface
2. Signal wall maintains interface regardless of power mode
   - Can force to a known value (e.g., the non-gated power rail)
   - Can perform level conversion

Block

Port A

Power control interface

Signal wall

Port B

Interface for a block with two ports

Example signal wall schematic (Port)

number of signaling ports to connect to other PDs. The signaling ports can contain a number of features such as level conversion or signal conditioning, as shown in the slide.

Example: PDs in Sensor Network Processor

2.7×2.7 mm² (130 nm CMOS)

Clock Rates 8–80 KHz
Supply 0.3–1 V
Leakage Power 53 µW
Average Power 150 µW
Peak Power 5 mW

Slide 10.59
A structured approach to the construction of PDs can also help to address the challenge of the proper conditioning of signals crossing power domains. Putting a wrapper around every PD supporting only standardized interfaces makes the task of composing a complex SoC containing many PDs a lot simpler. For instance, the interface of each PD should support a port to communicate with the PM through the “Power Network” and a

counter, whereas the others are implemented as dedicated hardware modules – as well as dissimilar schedules. It is rare for all functions to execute simultaneously. To minimize standby power (which is absolutely essential for this low duty cycle application), an integrated power manager assumes that any module is in power-down mode by default. Modules transition to active mode as a result of either timer events (all timers are incorporated in the PM), or events at their input ports. For a module in standby, the supply voltage is ramped down either to GND if there is no state, or to a data retention

Slide 10.60
An example of a structured power management approach is shown on this slide. This integrated protocol and application processor for wireless sensor networks combines a broad range of functions such as the baseband, link, media-access and network-level processing of the wireless network, node locationing, as well as application-level processing. These tasks exhibit vastly different execution requirements – some of them are implemented in software on the embedded 8051 micro-con
voltage of 300 mV. The latter is the case for the embedded micro-controller, whose state is retained in between active modes. To minimize overhead, the retention voltage is generated by an on-chip voltage converter. When the chip is in its deepest sleep mode, only the PM running at an 80 kHz clock frequency is still active.

The logic analyzer traces show how all modules are in standby mode by default. Power is mostly consumed during a periodic RX cycle, when the node is listening for incoming traffic, or during a longer TX cycle. Modules servicing the different layers of the protocol stack are only fired up when needed. For instance, it is possible to forward a packet without waking up the micro-controller.

---

**Integrated Switched-Capacitor Voltage Converter**

For low-power applications, such as wireless sensor networks, using off-the-shelf components to generate the various voltages that are needed on the chip turns out to be very inefficient. Most commercial voltage regulators are optimized for high-power applications drawing Amperes of current. When operated at mW levels, their efficiency drops to the single-digit percentage level (or even lower). Hence, integrating the regulators and converters on-chip is an attractive solution. The fact that the current demands for these converters are very low helps substantially in that respect. An additional benefit of the integrated approach is that the operational parameters of the converter can be adapted to the current demand, maintaining a high level of efficiency over the complete operation range.

The “switched-capacitor” (SC) converter, shown in this slide, works very well at low current levels, and can be easily integrated on a chip together with the active circuitry (such as in the case of the sensor-network processor of Slide 10.60). No special demands are placed on the technology. The ripple on the output voltage is determined by the current drawn (represented by the load resistor $R_{load}$), the total capacitance in the converter, and the clocking frequency. During the standby mode, the load resistance is large, which means that the clock frequency of the converter can be reduced substantially while keeping the voltage ripple constant. Hence, high levels of efficiency can be maintained for both active and standby modes. The only disadvantage is that the capacitors composing SC converters consume a substantial amount of silicon area. This makes their use prohibitive for applications that draw a substantial amount of current. Advanced packaging technologies can help to offset some of these concerns.

---

**Slide 10.62**

Using the SC-converter concept, it is possible to create a fully integrated power train for wireless sensor applications. As was mentioned in the introduction chapter (Chapter 1), distributed sensor
network nodes strive to harvest their energy from the environment to ensure operational longevity. Depending upon the energy source, rectification may be necessary. The scavenged energy is temporarily stored on either a rechargeable battery or a supercapacitor to balance supply and demand times. The sensor node itself requires a variety of voltages. Sensors, for example, tend to require higher operational voltages than digital or mixed-signal hardware. A bank of switched-capacitor converters can be used to provide the necessary voltage levels, all of which need to have the possibility to be ramped down to zero volt or the DRV for standby.

A dedicated integrated power-conversion chip, accomplishing all these functions for a wireless sensor node targeting tire-pressure monitoring applications, is shown in this slide. The IC contains the rectifiers as well as the various level converters. High levels of conversion efficiency are maintained over all operational modes.

**Slide 10.63**

Unfortunately, SC voltage converters are only effective at low current and power levels (i.e., at the mA and mW range). Most integrated circuits run at substantially higher current levels, and require more intricate voltage regulators and converters. The most efficient ones are based on resonant LC networks (most often called buck converters), where energy is transferred with minimal losses between an inductor and a capacitor at a well-defined switching rate. An example of such a converter is shown on the slide.
LC-based voltage regulators are generally implemented as stand-alone components. There is a very good reason for this: the required values and quality factors of the inductors and capacitors are hard to accomplish on-chip. Hence, the passives are most often implemented as discrete components. For SoCs with multiple power domains and dynamically varying voltage requirements, there are some compelling reasons to strive for a tighter integration of the passives with the active circuitry. Direct integration of the controller circuitry with the load leads to more precise control, higher efficiencies, and increased flexibility.

Though integrating the Ls and Cs directly on the IC may not be feasible, an alternative approach is to implement the passives on a second die (implemented on a substrate of silicon or some other material such as a plastic/glass interposer), which provides high-quality conductors and isolators, but does not require small feature sizes. The dies can then be connected together using advanced packaging strategies. An example of such a combined inductor/capacitor circuit is shown. Capacitance and inductance values in the nF and nH range, respectively, can be realized in this way. The concept of stacking dies in a 3D fashion is gaining rapid acceptance these days—driven mostly by the size constraints of mobile applications. This trend surely plays in favor of closer integration of the power regulation with the load circuitry, and of distributed power generation and conversion.

The possibility of multi-dimensional integration may lead to a complete rethinking of how power is distributed for complex SoCs. In the ICs of the past years, the power distribution network consisted of a large grid of connected Copper (or Al) wires, all of which were set to the nominal supply voltage (e.g., 1 V). The concept of power gating has changed this practice a little: instead of being connected directly to the power grid, modules now connect through switches that allow an idle module to be disconnected.

If it becomes possible to integrate voltage converters (transformers) more tightly into the network, a totally new approach may arise. This would resemble the way power is distributed in large scale at the metropolitan and national levels: the main grid is operated at high voltage levels, which helps to reduce the current levels and improves the efficiency. When needed, the power is down-converted to lower levels. In addition to the introduction of transformers, switches also can be introduced at multiple levels of the hierarchy.
Revisiting Power Distribution

A graphical representation of the constructions this vision could lead to is shown. Most standard circuitry is implemented on the base chip. Also included on the die is the control circuitry for the power regulators of the various power domains. However, the power grids of the latter are not connected on the die. The “higher levels” of the power distribution network are implemented on an interposer die, which implements a grid of high-quality inductors and capacitors, as well as a high-voltage power grid. The 2.5D integration strategy also allows for non-traditional technologies such as MEMs, or non-digital technologies such as DRAMs, to be tightly integrated with the computational fabric in a compact package.

Note: The term 2.5D integration relates to a three-dimensional IC technology, where individual dies are stacked on top of each other and interconnected using solder bumps or wire bonding. A true three-dimensional integration strategy, on the other hand, supposes that all active and passive devices are realized as a single artifact by constructively creating a stack of many layers deposited on top of one another. Although this ultimately may be the better solution, a large volume of economical and technological issues make the latter approach quite impractical for the time being.

Summary

- Power and energy optimality a function of operational parameters
- Runtime power optimization tracks changes in activity and environmental conditions to dynamically set supply and threshold voltages
- Aggressive deployment scales supply voltage below the traditional worst-case and uses error detection/correction to deal with rare errors
- Interesting idea: errors are not always fatal and can be allowed under certain conditions
- Challenge: Integrated power management and distribution supporting dynamic variations

In summary, the combination of variations in activity, process, and environmental conditions is leading to fundamental changes in the way ICs and SoCs are being designed and managed. Rather than relying solely on design-time optimizations, contemporary integrated circuits adjust parameters such as the supply and well voltages on the fly, based on observation of workload, leakage, and temperature. In addition, different parameter sets can be applied to individual regions of the chip called power domains.
This design strategy represents a major departure from the methodologies of the past. It challenges the standard design flows – yet does not make them obsolete. Actually, upon further contemplation, we can come to the conclusion that the idea of runtime optimization may make the traditional design strategies more robust in light of the challenges of the nanometer era, at the same time helping to reduce energy substantially.

### References (cont.)

<table>
<thead>
<tr>
<th>References (cont.)</th>
</tr>
</thead>
</table>