Power dissipation occurs when you turn on a processing element, and dynamic-power dissipation occurs during computations. You can represent the total power dissipation as the sum of the static power and the dynamic power. The static-power dissipation occurs even when no computations are occurring. Leakage power and bias power are the main contributors to static-power usage.

Dynamic-power dissipation results from short-circuit power and switching power. The short-circuit power consumption is proportional to the supply voltage. The switching power dissipates when the parasitic capacitors of the transistor gates charge and discharge during computational tasks. So the processing element’s power usage is equal to the sum of leakage power, bias power, short-circuit power, and switching power, where leakage power and bias power contribute to static power, and short-circuit and switching power contribute to dynamic power.

The switching power is the dominant source of power usage, accounting for approximately 90% of the total power that mainstream processing elements consume (Reference 1). The common equation for calculating the switching power of a processing element is:

$$P_{SW} = \alpha \times C_L \times V_{DD}^2 \times F$$

where $P_{SW}$ is the switching power; $\alpha$ is a constant representing the switching activity for the computational task; $C_L$ is the effective circuit-load capacitance, which you can assume to be a constant that the complexity of the design and the circuit technology determines; $V_{DD}$ is the supply voltage; and $F$ is the clock frequency (Reference 2).

This equation shows that you can reduce either the processing element’s frequency of operation or the supply voltage to reduce switching-power dissipation. Because switching power is proportional to the supply voltage squared,
you can achieve the largest energy savings by reducing the circuit’s supply voltage. In some cases, you can achieve further energy savings by also reducing the operation frequency. You must carefully apply this technique because it increases processing time. Because power dissipation over time determines energy dissipation, you may achieve no energy savings if extra processing time is necessary. However, careful application of both frequency and voltage scaling offers higher energy savings than voltage scaling alone. Reducing supply voltage also reduces leakage-power consumption, improving static-power dissipation (Reference 3).

Dynamic-power management uses standby or sleep modes to reduce power consumption. Because it takes time and energy to reactivate processing elements and buses, you should carefully apply dynamic-power management to ensure that no violations occur in the system’s operation or, in the worst case, no increases in power consumption from reactivation occur. Components still dissipate energy in standby as static power determines.

Dynamic voltage and frequency scaling reduces switching-power dissipation. This process can increase computation time, so you can apply dynamic frequency scaling only when there is slack time in the system-level operation of the design. Figure 1 shows an example of dynamic voltage and frequency scaling. In this case, the system schedule allows 30 msec for Processing Element 2 to complete a task. However, the processing element completes the task in 15 msec, leaving 15 msec of slack time. The supply voltage and operational frequency of this element decrease until the task completes in 30 msec to match the system’s schedule. Doing so decreases Processing Element 2’s power consumption.

You can also use dynamic-power management to maximize a design’s energy efficiency. Even if you apply dynamic voltage and frequency scaling to all components that are adapting their performance to the requirements of the system schedule and minimizing energy consumption, idle times may still occur. You can then use dynamic-power management to shut down components that are idle for a time for even further energy savings.

BUS ENERGY DISSIPATION

Communication between elements is essential in embedded systems with multiple processing elements. With every data transfer over a communication bus, the line capacitance charges and discharges, drawing current from the I/O pins of the elements. The following equation calculates the power that these currents dissipate:

\[ P_{CL} = b \times C_{BUS} \times F_{BUS} \times V_{TR}^2 \]

where \( P_{CL} \) is the power loss that these currents dissipate, \( b \times C_{BUS} \) represents the switched load capacitance of the bus, \( F_{BUS} \) is the operational frequency of the bus, and \( V_{TR} \) is the transmission voltage.

You can reduce the transmission voltage in communications buses only to a limit because of noise issues. Noise can more easily corrupt low-voltage communications, causing reliability problems. As with dynamic frequency scaling, you can scale down the operational frequency or data-transfer rate of the bus if the system schedule has slack time for bus communication. You can also put the bus into a standby state during idle times in an approach similar to dynamic-power management.

Low-power DDR (double-data-rate) DRAM devices and several popular communication buses offer low-power modes. These buses include PCIe (PCI Express), MIPI D-PHY and M-PHY, USB 3.0, and MXM (Mobile PCIe Module). Debugging these buses presents a number of challenges.

PCie’S LOW-POWER MODE

PCie specifications provide active-state power management that conserves power by putting the bus into a powersaving state or dynamically configuring the link’s width or speed.

Figure 2 A logic analyzer with a serial module for PCIe is the troubleshooting tool of choice because it allows you to trigger on a transaction-layer-packet-configuration write based on the bus, device, and function number.
lems can arise when the system enters or exits one of the power-saving link states or when the link’s width or speed changes dynamically in response to system requirements.

Table 1 (pg 36) lists the PCIe's link power states. To maintain synchronization between the transmitter and the receiver, the bus must transmit idle symbols over the link when there is no data available. The receiver decodes and discards these idle symbols. To save power during these periods, you can put the link into a power-saving state. The power savings and the time to recover back to the L0 state increase as the link moves from the L0 state to the L3 state.

To understand how this situation increases complexity, consider a case in which a PCIe link is in the L0 state and moves to the L0S state. Immediately after the transition, a transaction-layer-packet configuration write occurs that writes an incorrect value to a register, causing the system to crash. To troubleshoot this problem, you must acquire all the transactions that occur during the transition from the L0S state to the L0 state.

In such a case, a logic analyzer with a serial module for PCIe would be the troubleshooting tool of choice because it allows you to trigger on an event. For example, in this scenario, the logic analyzer would trigger on a transaction-layer-packet configuration write based on the bus, device, and function number (Figure 2).

After you define the trigger, the serial module can bit-lock and align the data across all the lanes of the bus after observing approximately 12 fast-training-sequence packets as the link exits the L0S state and enters the L0 state. Because the logic analyzer can track the change in the link state, it can acquire all transactions that occur immediately after the bus enters the L0 state, providing insight into the cause of the system crash.

To save memory, you can also set up the logic analyzer to filter out unwanted...
ed data in real time, focusing data acquisition on problem areas. A common use of filtering occurs when the PCIe bus is in the idle state. You can define a filter to filter the idle symbols in real time, storing only the required data and thus making more efficient use of the logic analyzer’s memory and capturing more relevant data that helps in troubleshooting the problem.

The PCIe specification also provides for dynamically changing the link’s width or speed depending on the need to conserve power or provide performance. These dynamic changes in the link’s condition can be challenging to debug.

Consider a case in which the link’s width is changing from eight lanes to four lanes. Here, you can use a logic analyzer to trigger on and acquire the training sequences that occur during the link-speed change and the link-width negotiation process, allowing you to validate that the link is training to the correct width. Figure 3 shows the type of dialogue you use to set up the condition for triggering on a training sequence. In some cases, due to errors in the link, you may not find the required trigger conditions. In such instances, you may need to build a customized sequence and set it as a trigger condition on a lane. This approach is faster than manually looking through the data to figure out the problems in the link. To identify errors in the physical layer, logical analyzers offer link-event triggers. For this trigger, the event could be disparity, an 8/10b error, or an error in framing the data-link-layer or the transaction-layer packets.

LOW-POWER DDR MEMORY, OR MOBILE DDR, OPERATES AT 1.8V RATHER THAN THE MORE TRADITIONAL 2.5V.

Low-power DDR memory, or mobile DDR, helps reduce energy requirements by providing more efficient device operation. It operates at 1.8V rather than the more traditional 2.5V. Low-power DDR DRAM commonly finds use in portable electronic devices, and line-powered electronics are increasingly adopting it as a way to reduce energy requirements.

Reducing operating voltage is a trend that extends beyond low-power DDR memory to more mainstream memory technologies, as well. DDR2, which originally operated at 2.5V, has seen later variants that lower the requirement to 1.8V, and further reductions are in development. Similarly, DDR3 once operated at a supply voltage of 1.5V but will soon see that figure decrease to 1.35V for some new components. Low-power DDR2, the newest entry in this power-
reduction trend, requires only 1.2V.

You can achieve additional energy savings by reducing the performance of the device. Low-power DDR and other DDR standards specify power-saving modes of operation that reduce performance depending on the system’s needs.

POWER-SAVING MODES

Because DRAM cells leak off charge, they must regularly refresh their contents during modes of operation requiring maintenance of data. The low-power-DDR-DRAM specification calls for three refresh modes to minimize power dissipation and maintain the required data states. The most basic mode, self-refresh, generates a low-frequency internal clock to maintain the contents of the DRAM. Temperature-compensated self-refresh automatically modifies the internal refresh clock frequency depending on the temperature of the low-power DDR DRAM. During lower operating temperatures, the refresh time can be longer to save power. Partial-array self-refresh maintains data in only a portion of the DRAM.

When the low-power-DDR-DRAM device does not need to retain data and when access to the DRAM is not necessary for several seconds, the device can use the power-down mode.

A system’s power consumption is proportional to the frequency at which the clock is changing. The low-power-DDR-memory standard stipulates many power-saving modes that leverage the frequency component of this power equation. The power-saving refresh modes reduce the clock frequency to reduce power consumption. The power-down mode can put the DRAM into standby mode during inactive periods. All of these power-saving modes primarily affect static-power consumption.

You can reduce dynamic-power consumption by optimizing data throughput, allowing the operating frequency of the device to decrease and still meet performance requirements. The ability to do this task is a key differentiator of low-power-DDR-DRAM devices.

LOW-POWER DDR DRAM

JEDEC (Joint Electron Device Engineering Committee) has specified the jitter, timing, and electrical-signal-quality tests for validating memory devices. The JEDEC specifications describe a comprehensive set of tests for each memory technology, including parameters such as clock jitter, setup-and-hold timing, signal overshoot, undershoot, and transition voltages. These specified tests are not only numerous but also complex to measure using general-purpose tools.

An example is measurement reference levels. JEDEC specifies certain voltage reference levels that you must use when making timing measurements. Figure 4 shows the ac and dc high and low input-voltage levels that timing measurements on data signals use. JEDEC defines the levels for rising and falling edges differently. Because of the complexity inherent in the JEDEC-specified measurement methods, including reference levels and pass/fail limits, the preferred approach is to use an application-specific measure-
ment utility for DDR test. Using such a utility ensures that you configure measurements according to the specification and reduces setup time.

With real-time-performance oscilloscopes, DDR software utilities provide a broad set of measurements that conform to the JEDEC specifications. In addition, these utilities allow you to customize many settings to accommodate measurement tasks on nonstandard devices or system implementations and to aid in debugging. To ease setup tasks, a menu-driven interface guides the user through a selection process (Figure 5).

The first step of such an interface is to select the DDR generation you want to test and the speed grade of the memory. In addition to the default choices, the use of custom speed settings makes the software adaptable to future technology advances, overclocking applications, and the like. Once you have selected the generation and data rate, the software configures the correct voltage references for measurements. The next step is to select which measurements to perform (Figure 6). The menu groups available measurements according to which signals and probing connections are necessary. The remaining steps guide you on how to probe the needed signals and offer additional opportunities for customizing or adjusting parameters, such as measurement reference levels.

Once the setup is complete, the oscilloscope acquires the signals of interest, identifies and marks data bursts if needed, and makes the selected measurements. A results panel shows all measurement results with statistical population, spec limits, pass/fail results, and other data (Figure 7). You can at this point print a report, with an option to also save the waveform data that you used to make the measurements.

Because the captured waveform data is available with the measurement results, you can use this information for further analysis. For example, if a measurement fails the spec limits, you can identify exactly where in the waveform record the failure occurred and then zoom in on the region of interest to investigate the exact signal details and characteristics at the time of failure.

System-level energy-management techniques look at the system-level operation of a design for opportunities to lower power consumption by shutting down components or scaling voltage and frequency. Test-and-measurement tools have evolved to help designers debug systems in the face of this increased complexity.

For instance, logic analyzers offer trigger capability with a layout similar to the definitions in the standard they are testing—a helpful technique in finding elusive problems resulting from active-state management. Similarly, validating DDR-DRAM devices requires performing the numerous tests in the JEDEC specifications, a time-consuming and complicated task. By using specialized software together with a high-performance real-time oscilloscope, you can access a broad set of automated measurements, simplifying the validation of memory devices.

**TABLE 1 PCIe LINK STATES**

<table>
<thead>
<tr>
<th>PCIe LINK STATE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>L0: active state</td>
<td>All transactions are enabled, and link is operating in normal mode</td>
</tr>
<tr>
<td>L0S: low resume latency, energy-saving standby</td>
<td>All power supplies and reference clocks are active, transaction-layer-packet and data-link-layer packet transmission are disabled</td>
</tr>
<tr>
<td>L1: higher latency, low-power standby</td>
<td>Transaction-layer-packet and data-link-layer packet transmission are disabled</td>
</tr>
<tr>
<td>L1/L2: staging point for transition to L1 or L2</td>
<td>Main power supply and reference clocks are off</td>
</tr>
<tr>
<td>L2: auxiliary powered link, deep energy-saving</td>
<td>Link is in this state when no power is applied</td>
</tr>
<tr>
<td>L3: link off</td>
<td>All transactions are enabled, and link is operating in normal mode</td>
</tr>
</tbody>
</table>

**REFERENCES**


**AUTHOR’S BIOGRAPHY**

Gina Bonini is the worldwide embedded-system technical-marketing manager for Tektronix. She has for more than 15 years worked extensively in various test-and-measurement positions, including product planning, product marketing, and business and market development. She holds a bachelor’s degree in chemical engineering from the University of California—Berkeley and a master’s degree in electrical engineering from Stanford University (Stanford, CA).