Almost every system designer or integrator deals with power-consumption issues. One area of interest centers on the processor controlling the system. Data-sheet specifications serve as guidelines, but you’re on your own if you want to know exactly how that processor behaves in your system while running your application code. Processors with "high-tech" packaging and multiple power planes can make it difficult for you to make these measurements in situ.

For this article, engineers from ARM, IBM, Integrated Device Technology (IDT), Mitsubishi Electric, Motorola, and NEC mountain-biked with me up to the Tektronix facility in Beaverton, OR (group picture). Our goal was to measure the variables in the equation P=V*I, where P=power, V=voltage, and I=current. I asked each microprocessor-vendor participant to supply an embedded board with processor, memory, cables, and power supply. I also asked each participant to provide access to the processor's power-supply inputs and program-monitoring pins, which proved futile when a program fit entirely within cache. Tektronix supplied its state-of-the-art equipment, engineers, and technicians to measure processor power consumption while running benchmarks from the EDN Embedded Microprocessor Benchmark Consortium (EEMBC).

Choosing the proper measurement tools

To move forward with this project, we had to consider:

- type of power consumption the design needed,
- total system power and processor-only power consumption,
- measurement of steady-state or instantaneous power consumption,
- representative application code, and
- equipment to use.

The traditional way to measure power consumption involves the use of a DMM. A DMM allows you to
obtain a power reading by separately measuring the root-mean-square (rms) voltage and the rms current and multiplying the two figures. To make these measurements, you must run a DMM in series with the processor's input power supply. The sensitive processor circuits cannot tolerate the impedance from the DMM's loading and alter the processor's behavior. So, for this article, Tektronix rolled in its new TDS 694C digital storage oscilloscope (DSO) with the TCP202 dc-coupled current probe. The company also brought in a suite of active voltage measurement probes, including the 1.5-GHz P6245 and 4-GHz P6249 single-ended probes, and the 1-GHz P6247 and 1.7-GHz P6248 differential probes (see sidebar "Probing into the equipment"). The 50-MHz TCP202 probe has less than 7-nsec rise time, which provides ample accuracy for this project. The TCP202 can also measure both dc and ac of 10 mA to 15A as high as 50 MHz. Feeding both the voltage and the current measurements into the TDS694C enabled us to use the scope's ability to multiply the two measurements and display the power measurements we requested. The tricky part about using the current and voltage probes was finding a physical connection with the processor's power supply. IDT used its Orion 79R4640 or 79S440 evaluation-and-development platform for this project. This platform supports the company's 32-bit R4640 and 64-bit R4700; you can also run the 64-bit processor in 32-bit mode. One of the hardware jumpers on the board passes the supply current to the processor card, allowing us to easily remove the jumper and connect a wire loop between the two jumper posts. We then connected the current probe to the loop. The Tektronix manual suggests using #18 AWG wire and keeping the wire loop as short as possible. The bottom line is that you should minimize the impedance for the circuit under test.

Unfortunately, testing the other platforms for this project was more difficult. For example, IBM provided a combined motherboard/daughtercard with a PowerPC 750. The connector on the motherboard provides 21 3.3V pins to a power plane on the daughtercard. The power plane attaches to a voltage regulator, the Level 2 cache, and a programmable-logic device. To allow us to attach the current probe and measure the 3.3V supply, IBM application engineer Frank Frischauf had to desolder and electrically isolate all 21 V_{DD} pins, allowing us to connect a wire directly from the 3.3V supply on the motherboard to a bypass capacitor on the daughtercard. Frischauf also had to break the link between the 2V regulator and the processor core's input-voltage pin. A quick solder job allowed him to reconnect the link using a wire loop for the current probe (Figure 1).

### Measuring steady-state power

We also had to consider whether we wanted to measure the steady-state or the instantaneous power consumption. In CMOS logic, the internal node voltages swing from one power-supply rail to the other and consume power only when switching states. It shouldn't surprise you that CMOS current is directly proportional to the switching rate. However, power consumption is directly proportional to frequency only if you could run your program entirely from cache. External program and data fetches do not scale with the core frequency.

Program activity also affects current consumption. Complex instructions that involve multiple execution units, caches, or memory buses consume more current than simpler instructions. Theoretically, the TCP202 current probe has enough bandwidth to allow you to measure the device's current consumption on a per-instruction basis. However, numerous practical limitations exist. The most significant of these limitations is the need for bypass and filter capacitance on the power-supply lines to minimize spiking. The capacitors smooth any instantaneous transitions. Remove the capacitors, and good luck running any processor that runs any faster than 10 MHz. (I found an interesting application note from Texas Instruments, "Calculation of TMS320LC54x Power Dissipation." The note details the results of a test in which the company removed the capacitors and operated the device at 10 MHz.)

However, we left the capacitors intact. Furthermore, the steady-state measurement suffices for...
obtaining the typical and maximum power consumption. However, the instantaneous-power measurement would be useful if you were trying to squeeze every last drop of energy from a battery-powered application. This approach would allow you to develop a power profile for each instruction and tune your application code accordingly.

**Isolating the processor**

High-speed processors require multiple power inputs to reduce internal noise levels. This fact, along with the fact that the processors' packages have impossible-to-access pins, limited our ability to directly measure the power at the processor. Instead, with some platforms, we were able to carefully desolder the output line on the voltage regulators so that we could insert a wire loop for attachment of the current probe. (As noted, this situation doesn't apply to the IDT processor board.)

An issue still remains with measuring the I/O-related power of a processor because this power depends on the bus loading. Recall the equation \( I_{\text{LOAD}} = C_{\text{LOAD}} \times V_{\text{DD}} \times X_{F} \). So, you can vary the number of memory devices attached to the processor, which in turn varies the current and, therefore, the power consumed.

It's interesting to determine how close a processor vendor's data-sheet specifications for typical power consumption are to real-world measurements. To emulate the real-world behavior, we ran some of the benchmarks from the EEMBC suites as a representative workload ([www.eembc.org/benchmark.htm](http://www.eembc.org/benchmark.htm)). For the most part, it really didn't matter which of the EEMBC benchmarks we ran on a processor: Most of the benchmarks yielded the same results for steady-state power. On the other hand, the execution times varied, and those situations required that we convert power into energy.

Our goal was to use Tektronix's TLA 714 logic analyzer to monitor the address, data, or control lines of the target system to determine exactly when to start the acquisition system of the DSO. The TDS 694C contains a Java-based application interface that allows you to program the DSO to perform various mathematical functions. We used the instrument's math-function capability to take the acquired waveforms and calculate the average power, peak power, and energy the processor consumed.

IBM provided its PowerPC 60x/7xx/CPC700 evaluation kit with a PowerPC 750 daughtercard and 16 Mbytes of synchronous DRAM (SDRAM) ([Figure 2](#)). The 750 contains a floating-point unit (FPU), separate 32-kbyte Level 1 instruction and data caches, and an integrated Level 2 cache controller; the daughtercard contains a 512-kbyte Level 2 cache. IBM's debugger, RISCwatch, allowed us to turn the caches on and off and study the effects on performance and power consumption ([Table 1](#)). For example, we ran the integer version of the Bezier-curve calculation benchmark from EEMBC's Office Automation suite. Turning off the Level 2 cache did not degrade performance; however, the core's power consumption dropped 8.8%, indicating that the benchmark fit inside the Level 1 cache. When we turned off both the Level 1 and the Level 2 caches, performance dropped by a factor of 28, and power consumption dropped 36%. The power consumption decreases because of numerous CPU stalls waiting for data.

We also used the floating-point version of the Bezier-curve benchmark to compare the power difference associated with using the 750's FPU. With caches enabled, the floating-point version caused the 750 to consume an average of 8.8% more power. This increase was the result of the complexity of the FPU's circuitry and the 64-bit FPU register file, as well as the caches' higher demand for data.

We also observed the effect of changing the processor's clock frequency. When operating at 266
MHz and 1.8V, the 750's core consumed 50% less power but ran 34% slower than when operating at 400 MHz and 2V for the PATRICIA (Practical Algorithm to Retrieve Information in Alphanumeric) networking benchmark (Table 1). The Bezier-curve benchmark produced roughly the same results as the PATRICIA benchmark. If energy consumption is one of your prime considerations, you might be willing to give up performance and run at a lower frequency and lower voltage. Also note from the table that the power consumption of the I/O and L2 cache increased when we decreased the operating frequency and voltage. This result is puzzling. If you have any suggestions on why this situation occurs, write me at markus@embedded-benchmarks.com.

One last point worth making about the IBM device is that the data-sheet specifications for the 400-MHz 750's typical and maximum core power are 4.7 and 6W, respectively. These figures are roughly 20 to 30% higher than the experimental values we obtained. As you analyze this data, keep in mind that IBM's specification represents an average that the company determines with worst-case process parts, a worst-case instruction mix, and worst-case I/O cycling (such as switching between 0Ahs and 05hs on the data bus). This project shows that power consumption can vary significantly with cache configuration, system-bus loading, and application code. Thus, it is essential to characterize the power consumption of the target system rather than relying on specs.

Chuck Corley, PowerPC application engineering manager for Motorola, came to the Tektronix facility with his PowerPC Excimer board. This board supports the PowerPC 603e processor with separate 16-kbyte instruction and data caches, 1 Mbyte of 3-1-1-1 SRAM, and 4 Mbytes of flash (Figure 3). Similar to the approach we used with the IBM part, Corley ran the Patricia benchmark using 133- and 266-MHz configurations of the 603e. The technical specifications of this device indicate that the 133- and 266-MHz configurations yield typical power consumption of 1.6 and 3.5W, respectively. Further, the maximum specified power ratings for the 133- and 266-MHz configurations are 2.4 and 5.3W, respectively. These numbers correlate closely with our values, which we measured at the core's power supply, and include 15 mW more for the PLL and 25 mW more for an indicator LED. We discovered that, operating at 133 MHz, the 603e consumed 2.22W—more than half the power that the 266-MHz configuration consumed at 4.09W. The 133-MHz configuration has two internal clocks for every external bus clock. In some cases, the benchmark can keep the execution units busy while waiting for memory; hence, fewer stalls and proportionately more power consumption results. However, when the system operates at 266 MHz, it has four internal clocks for every external bus clock, thereby increasing the odds that more stalls will occur. This situation causes a relative increase in power consumption at the lower operating frequency. You exacerbate this problem by turning off caches (Table 2).

To examine this phenomenon, we could have used the logic analyzer to trigger on the processor stalls. This approach would have allowed us to measure the instantaneous power consumption during each bus clock. Alternatively, we could have also measured the number of stalls during the benchmark and derived a correlation.

Brad Suessmith, ARM application engineer, arrived on the scene with his company's new Integrator evaluation board, which contains a 160-MHz, 3.3V ARM920T running with a 40-MHz bus clock. The 32-bit ARM920T RISC processor includes Thumb's 16-bit compressed instruction support and separate 16-kbyte instruction and data caches. The Integrator board contains 256 kbytes of SRAM and 16 Mbytes of SDRAM.

Suessmith configured the ARM platform to run various benchmarks from EEMBC's Automotive suite. Unfortunately, we were unable to use the logic analyzer to trigger the start of the benchmark (and start capturing data on the DSO) because the board lacked a physical connection for the analyzer. Instead, we set up the DSO to start acquiring data when it detected a change in power-consumption levels as the benchmark started. However, it was sometimes difficult to differentiate between a
noise spike and the benchmark's starting—a problem that the logic analyzer would have solved. The measurement represented the combined power from both the processor core and the I/O. Obviously, bus loading affected our measurement.

On several occasions we ran the same test but varied only the oscilloscope's sampling rate. For example, when running the Table Lookup benchmark, we used 2.5G- and 10G-sample/sec rates, which yield average power consumption of 420 and 388 mW, respectively—an 8% difference (Table 3). This difference may have occurred by coincidence. Alternatively, this difference may have occurred because we had reduced the time window on the DSO by a factor of four; thus we were recording a time during which the processor consumed more power. This greater-power-consumption scenario means that we could have spent more time analyzing shorter periods to obtain more accurate results. Shorter periods would be valuable for instantaneous- or peak-power measurements that spikes during switching cause. Longer records would be useful for maintaining high sample rates over a longer time window. This approach would have ensured that we captured all variations during the project's duration. The good news is that our measured values approach ARM's estimated specifications of 400 mW, or 2.5 mW/MHz; however, this specification represents only the core's power. (For more information, see www.arm.com/Pro+Peripherals/Cores/ARM9TDMI/flyer/table.html.)

Next came IDT application engineer Kasi Chopperla with IDT's Orion 79R4640 evaluation-development platform, which is similar to the ARM platform in that it lacks a physical connection to the Tektronix logic analyzer. Also similar to how the ARM platform works, IDT's measurements represent the combined power from both the processor core and the I/O. We set up the DSO with a 5-sec window to allow us to determine where the current changed as the benchmark program started. (In this case, we used EEMBC's Networking PATRICIA route-table-look-up benchmark.)

Before starting the benchmark (and with IDT's 79RC64475 processor running at 200 MHz), the processor's steady-state power was 1.08W at 3.3V; during this idle period, the processor was executing a debug-monitor program and polling for keyboard input. When we ran the benchmark, the average power rose to 3.01W and ran for about 4 sec. After we switched off the 79RC64475's cache, power dropped to an average of 2.03W, but performance dropped by almost a factor of 10.

**NEC cranks up its V832**

NEC engineer Bin Takahashi participated at Tektronix's facility with an RTE-V832-PC evaluation board. The board comes with a 143-MHz V832, 512 kbytes of SRAM, and 32 Mbytes of DRAM. The 32-bit V832 RISC processor has separate 8-kbyte, direct-mapped instruction and data caches. The V832 core operates from a 2.5V supply, and its I/O operates from 3.3V. For our measurements, we isolated both supplies and inserted a wire loop to attach the current probe.

We used the table-look-up and angle-to-time benchmarks from EEMBC's Automotive/Industrial suite for our power analysis. Although the RTE-V832-PC board lacked a hookup for the logic analyzer, Takahashi had configured Green Hills Software's (www.ghs.com) Multi debugger, allowing us to set a breakpoint at the start of the benchmark. To establish a baseline power level, we measured the 3.3 and 2.5V power consumption with the processor in breakpoint mode. The values were 151 and 16 mW, respectively. Essentially, these figures represent the power the processor consumes while the debugging ROM monitor is polling for data.

After starting the table-look-up benchmark, the power for the 3.3 and 2.5V supplies went to 107.9 and 17.6 mW, respectively. Note that the 2.5V supply power went up, whereas the 3.3V supply power went down. NEC accounts for the difference by stating that, during the breakpoint mode, the processor executes from SRAM, and, during the benchmark test, it executes from the lower power
SDRAM.

Mitsubishi Electric provided Application Engineer John Taft and Engineering Manager Jeff Canion, along with the company's PC4701HS emulator and M30620T pod to control the controller. The device under test, the MSV3062ECFS, a 16-bit M16C 62 series, operates at 10 MHz. We were unable to hook up a logic analyzer to the Mitsubishi board. This problem prevented us from setting up a trigger for the logic analyzer to signal the start of data acquisition. Alternatively, we set the trigger level on the oscilloscope to begin acquiring data after it detected a sharp increase in current (as the processor went from idle to executing the benchmark). The data sheet for this device indicates typical and maximum power ratings of 95 and 190 mW, respectively. While running the PWM benchmark from EEMBC’s Automotive suite, our test measurements reached a steady-state value of 1.6W and a high of 1.9W (Figure 4). Mitsubishi was unable to account for the discrepancy between our measured values and its printed specifications.

If we could do it again

If we had the opportunity to repeat this project, we would have found it worthwhile to increase the sampling rate on the oscilloscope so that we could have generated more detailed measurements. (If you increase the sampling rate, you must increase the record length to equal the measurement period.) It would also have been helpful if we could have better used the logic analyzer by having been able to access some program-monitoring signals. This approach would have allowed us to isolate segments of code execution and even to have monitored the effects of cache misses and other functions that would cause the power consumption to deviate from steady state.

If you routinely measure power, design the processor board so that the processor has a power-supply section. This process lets you measure processor power without determining system power. Design the processor power-supply leads so that you can open them and easily insert an ac- or ac/dc-measuring probe. An ac/dc probe works better for determining the overall current, the instantaneous current, and the steady-state currents. The current probe should also have a bandpass filter to capture all relevant signal deviations.

You should try to measure the processor's current at the component side of the decoupling capacitance. Use low-capacitance, low-induction, and high-resistance measuring tools that provide the lowest device-under-test loading.
Standardized power measurements simplify µP evaluation

By Jih-Sheng (Jason) Lai, PhD, Associate Professor, Center for Power Electronics Systems, Virginia Polytechnic Institute

Working on EDN's hands-on project at the Tektronix campus in Beaverton, OR, was revealing and interesting.

It quickly became apparent how necessary it is to have a benchmark-evaluation approach to power measurements. With the ongoing interest in energy conservation and the concerns about excessive heat, more system designers must assess the power efficiency of their microprocessor-based applications. Without benchmarks, it is difficult to evaluate competing architectures' power consumption. The benchmark evaluation provides a straightforward, quantitative way to evaluate the energy efficiency of competing architectures.

Here are some hints on how to obtain useful power measurements when using these power definitions. First, carefully define the optimal time period around which to make the power measurement. Instantaneous output power can sometimes be higher than instantaneous input power because of an energy-storage element in the system. A possible result of such a situation is that the average output power could be larger than the average input power—an obviously invalid result. For this reason, you should avoid these time periods or define a sufficiently long period when analyzing the power activity.

Another crucial detail is to synchronize the input and output periods to get the right ratio for determining power consumption. Even if you optimize the measurement time period, you must synchronize the measurement period of input and output signals. Otherwise, the measured efficiency could once again be much higher than one, invalidating the measurement.

As design challenges become more complex, the need to understand power management increases. Yet, EEs are receiving less high-level education in power-related subjects, often learning only basic power and energy concepts. In addition, most designers have not handled power measurements since their college days. Fortunately, recent developments in power-measurement capabilities with some general-purpose oscilloscopes ensure the production of useful, reliable power data with visualized voltage and current waveforms. For example, Tektronix offers a Java-based power-measurement-software package in all of its 500/600/700 oscilloscopes, including its 3-GHz TDS 694C. This software option makes it easy for the designers to make sophisticated, accurate, and reliable power measurements. Time-domain voltages and currents are visible for calibration to the measurement period, and accurate synchronization is now possible (see sidebar “Probing into the equipment”).

Using this software to make the standardized power measurement and keeping in mind these tips, mainstream microprocessor designers can quickly produce reliable power results. These results, in turn, will enable them to easily compare microprocessor architectures' power, resulting in more effective energy efficiencies for their designs.
Probing into the equipment

Analyzing the power behavior in microprocessor-based designs requires a logic analyzer working with an oscilloscope, both with sufficient bandwidth for high clock rates. For this hands-on project, Tektronix Inc provided a suite of instruments, including the TLA 714/720 portable and benchtop logic analyzers; the TDS 694C digital storage oscilloscope (DSO); and several probing instruments, including the P6245, P6247, P6248, and P6249 (Figure A). Tektronix designed these instruments to work together to provide specialized features, such as time-aligned cross-triggering, to the device under test.

Our lab was stocked with the Windows 98-based Tektronix TLA 714 portable and TLA 720 benchtop mainframes and six logic-analyzer modules with prices beginning at $15,500. The modules provide as much as 16 Mbytes of memory and hardware-accelerated search-and-display features for faster information access. The logic analyzers' modularity allowed us to configure these instruments for added performance whenever needed. For instance, some of the participants upgraded the logic analyzers from 100 to 200 MHz of state speed, and others configured the instruments' memory at 64 kbytes to 16 Mbytes.

The TLA 714/720 provides as many as 680 channels; you can merge 408 of these channels for working with next-generation microprocessors or multibus applications. All support Tektronix's MagniVu acquisition technology, providing each logic-analyzer module with 500-psec timing resolution on all channels simultaneously—extremely helpful when we tried to isolate transient power behavior. This technology also allows the analyzer to provide 200-MHz synchronous acquisition and a 400-MHz typical data rate. The TLA 714 and 720 also provide simultaneous 200-MHz state and 2G-sample/sec timing on the same probe.

Tektronix provided $37,995 TDS 694C DSOs to complement its logic analyzers. These DSOs provide 3-GHz single-shot bandwidth on all four channels simultaneously. The combination of this high-analog bandwidth with a simultaneous 10G-sample/sec sample rate on all channels and a high-stability timebase allows signal-timing measurements as fast as 15 psec.

In addition to its general measurement and statistics capabilities, the TDS 694C offers jitter- and timing-analysis measurements through embedded Java software implementations, which are available on floppy disks. From the oscilloscope's menu-based interface, this application provides automatic measurements and analysis of clock timing in industry-specific terminology, automating such crucial measurements as cycle-to-cycle jitter, N-cycle jitter, skew, period, and duty cycle. One of these Java-enabled-measurement-software packages, TDSPWR1, provides custom power measurements and helps design engineers concerned with the power consumption of components, subassemblies, or systems. This package includes the calculation of instantaneous power and energy curves from voltage and current input. Automatic measurements include true power, apparent power, and power factor. And, because the oscilloscope is Java-based, you can customize it for more esoteric applications, such as three-phase measurements.

The seamless interaction between the TLA 714 and the TDS 694C allowed us to quickly capture the power activity we wanted to examine. First, we used the TLA 714/720 logic analyzers to find the processor sequence of interest. Then, the logic analyzers automatically cross-triggered the TDS 694C oscilloscope. Tektronix designed the two instruments to yield zero cross-triggering delay. Be careful here: If you use a different logic analyzer, the data may not be time-aligned, and you must deskew the information. As a result, we were confident that the same signal activity appeared on both the logic analyzer and the oscilloscope.

Probing was an important factor in our power-activity measurements. For those designs that had non-Earth-referenced power supplies, we used the $4295, 1.7-GHz Tektronix P6248 active differential probe. A differential probe allows you to measure signals that are not referenced to Earth ground and reduces the amount of common-mode signals or noise that may occur. Its bandwidth enabled us to capture and measure the fastest differential rise times. (The system bandwidth must be three to five times higher than the highest signal frequency of interest.) Equally important, the P6248 is a small probe and allows signal acquisition in tightly packaged circuits—exactly what we needed to attach to these microprocessors. The P6248 has two attenuation settings for testing versatility: 1X, providing 1.7 GHz, and 10X, providing 1.85 GHz (1X and 10X—typical probe only).

We used the $3395, 4-GHz Tektronix P6249 single-ended active probe for some systems that had their power supply referenced to ground. The 4-GHz P6249 is also a small probe that allowed us to capture and measure fast rise times. It has a dynamic range of 3.5V, and, therefore, you can use it on 3V or lower voltage logic—another important consideration for some of the participants.
Author info

You can reach Technical Editor Markus Levy at 1-916-939-1642, fax 1-916-939-1650, e-mail markus@embedded-benchmarks.com.