Technical Article **Step 6 to Build a Smart Thermostat Using an MCU: Energy Optimization**



Britta Ruelander

Co-authored by Bargain Nisarga, Systems Engineer at TI

In the first five installments of this series, we've helped you build a fully featured and network-connected smart thermostat. In this installment, we will focus on how you can best optimize power and energy consumption to extend the battery life of your thermostat.

A battery-powered wireless thermostat gives you "peel-and-stick" flexibility to use the product anywhere, even in the absence of pre-wiring for power or communication. However, if you don't design your thermostat to have an extended battery life, users will have to change the batteries often, which will affect your product's overall cost-effectiveness and convenience factor. Let's take a look at how to address this challenge.

Here are some terms we'll use in the context of power savings in this blog post:

- **Current consumption** refers to the current consumed by a module/device when operational, and is represented in amperes.
- **Power consumption** refers to the current consumed at a specific supply voltage level. Power = I × V and is represented in watts. This term is also interchangeably used with current consumption.
- Energy consumption depends on three factors: voltage, current and time. Energy = I × V × time. The total energy consumed includes the average current consumption based on the ratio of the time period during which the module/device is enabled or disabled, and the respective currents during those periods.

When you look at a microcontroller (MCU) data sheet, you might be startled – the current numbers may just seem too high for battery-powered applications. But here is how you need to read the numbers: the current numbers specified in a device data sheet show the current consumption assuming that the module is running with a 100% duty cycle. For an analog-to-digital converter (ADC), this means that you are sampling and converting data continuously, this is not realistic for most "real world" applications.

As we discussed in the second installment of this series, you will normally have longer intervals in between two samples; during this interval, the ADC will not be active. Because of its low duty cycle, the ADC will not consume much energy. So to calculate the average current consumption for all modules used in your application (like the central processing unit [CPU], ADC and comparator), you need to calculate the ratio between the on and off time periods and accordingly apply the current numbers specified in the data sheet. That's how you will calculate the average current consumption. The maximum consumption at a given instant determines the peak current consumption. Normally, the battery will determine the peak-current limit for your application.

Most low-power MCUs offer different power modes to reduce the application's overall power and energy consumption, typically supported by the following device power modes: active, sleep, deep sleep and shutdown mode. The power modes differ from each other depending on the clock sources available and the modules powered in that specific power mode. In general, lower power consumption in a specific power mode helps keep lesser functions functional and lowers the speed at which the modules operate.

Using MCU Power Modes to Optimize Current Consumption

Let's take a look at what techniques are used in different device power modes to enable power savings:

• Active mode has highest current consumption compared to other power modes, because the on-chip CPU is active. The faster the CPU frequency of operation, the higher the current consumption; therefore, a microampere/megahertz (uA/MHz) parameter represents this current number. On-chip peripherals typically

1



support clock-gating techniques to disable transistor switching within the peripheral circuitry when it is not in use or not enabled, and in turn enable current savings.

- In **sleep mode**, the high-frequency on-chip clock oscillators remain running. Clock-gating only the CPU enables it to resume executing instructions on the next clock cycle following a wake-up trigger.
- In **deep-sleep mode**, the high-frequency on-chip clock oscillators are disabled; a low-frequency clock is enabled (if needed) to keep a few critical peripherals running, such as the real-time clock (RTC) or the interval timer. Upon a wake-up trigger, it takes a few microseconds to re-enable the oscillators before resuming active or sleep-mode operation.
- Deep-sleep and shutdown modes use power-gating techniques to power down most parts of the device and keep only a few peripherals (or groups of peripherals) powered to achieve sub-1µA-range device current numbers.

Keep in mind that the typical current numbers in a device data sheet are specified for operation at room temperature. Current consumption at higher temperatures (especially for sleep and deep-sleep modes) may be much higher compared to typical numbers due to higher leakage current at the transistor level. You will need to consider the maximum current numbers in the device data sheet when calculating current consumption over the product's operating temperature range.

You should plan to optimize power consumption overall by grouping the on-chip peripherals that the device needs at different points of operation, and use the various power modes to power and clock only those modules that need to be active at a given instant. For example, when your smart thermostat is not actively measuring temperature, updating the liquid crystal display (LCD) or sampling the microphone input, the device can enter deep-sleep mode, where only the low-frequency clock is enabled for operations such as the RTC, periodic interval timekeeping (for temperature measurement) and capacitive touch sensing.

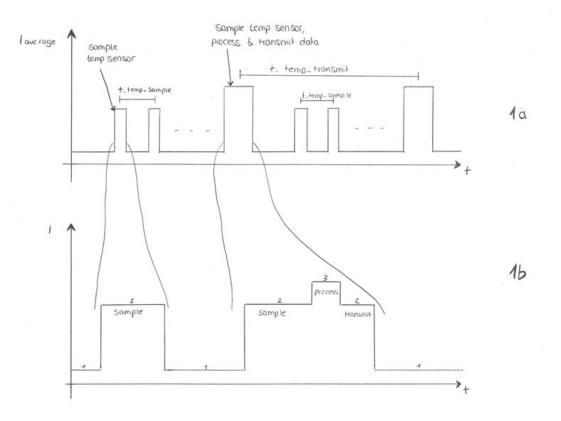


Figure 1. A Shows an Application-level Duty Cycling Profile for a Smart Thermostat Where the Temperature Is Sampled Every T-temp-sample; Samples Are Processed and Data Is Transmitted over the Network Every T-temp-transmit. Increasing T-temp-sample and T-temp-transmit at the Application Level Reduces the Overall Power Consumption.

2



Figure 1b shows the corresponding device level power profile capturing device operation in various power modes including:

- Deep-sleep mode when waiting between temperature samples.
- Sleep mode when the ADC is sampling and direct memory access (DMA) is moving the results to internal memory.
- Active mode when the CPU is processing data.

You can also incorporate power monitoring and optimization at the application level. Let's assume that you are measuring the temperature every few seconds and sending the measured values every few minutes. Along with this, you can add a battery monitor using the on-chip ADC peripheral to monitor the battery voltage level. When the battery voltage drops below the monitor threshold, the application can reduce the frequency of temperature sampling and communication over the network. Better still, the application can send a message to the host indicating a low battery level so that the user can change or charge the battery.

Tools That Help You Save Power

Ultra-low power MSP430[™] and MSP432[™] MCUs enable you to design your application power consumption to within your desired limits. Additionally, MSP430[™] and MSP432 MCUs offer useful tools to optimize your application for ultra-low power operation.

Both new and advanced embedded system developers may find these tools valuable:

- The ultra-low power (ULP) advisor is a tool for guiding developers to write more efficient code in order to fully utilize the unique low-power modes and features of MSP430 and MSP432 MCUs.
- EnergyTrace[™] software for MSP430 and MSP432 MCUs is an energy-based code analysis tool that measures and displays the application's energy profile and helps optimize for ultra-low-power consumption.

Here are the key takeaways from the sixth blog post of this series:

- MCU low-power modes help you reduce overall device current consumption by gating clock and power to modules that are not required to be functioning at all times during application operation.
- Duty-cycle the different MCU peripherals according to your needs and power budget, and make use of the MCU power modes accordingly to optimize the overall energy consumption of your smart thermostat.
- Leverage MSP430[™] and MSP432[™] power- and energy-optimization tools to fully utilize its low-power features.

Stay tuned for the final part of this blog series.

Additional Resources

 For more guidelines about calculating overall device power consumption and estimating the battery life of a specific battery, download the white paper, "Benchmarking MCU power consumption for ultra-low-power applications."

3

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to TI's Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265 Copyright © 2023, Texas Instruments Incorporated