Low Power Design Basics

How to Choose the Optimal Low Power MCU for Your Embedded System

by Mike Salas, MCU Marketing Manager
Silicon Laboratories Inc.

Saving energy is beneficial for the environment and easier on the pocketbook. The myriad benefits of improving energy efficiency have been well-documented: lower electric bills for consumers, reduced load on utilities, reduced cost of ownership for electronics products and fewer spent batteries thrown away in landfills.

As the use of electronic devices pervades virtually every aspect of our lives, reducing power consumption must start at the semiconductor level. The power-saving techniques that are designed in at the chip level have a far-reaching impact. This is especially true with regard to the microcontrollers (MCUs) that serve as the intelligent engines behind a majority of today's electronic devices.

From a systems architecture perspective, the challenge of identifying which MCUs truly are "low-power" requires designers to navigate though the myriad claims made by various semiconductor vendors. Because of the different (and often confusing) metrics used by vendors, this is not a simple task.

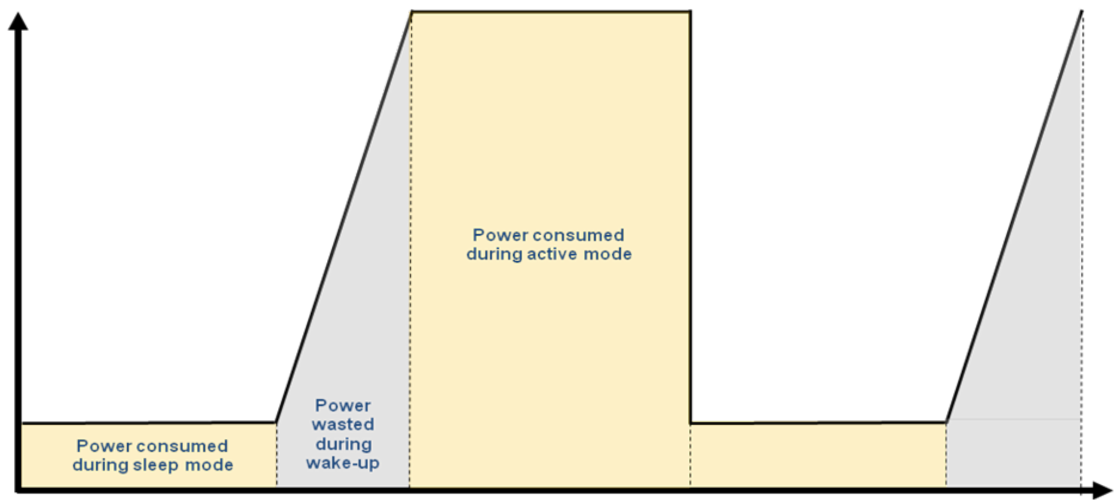
Let's take a closer look at the key factors that should be considered when analyzing the power efficiency of competitive MCU alternatives.

At a basic level, MCU power consumption can be defined as the sum of the following:

Total power consumed = Active mode power + Standby (sleep) mode power

However, another important metric to keep in mind is the amount of time it takes for an MCU to transition from a standby state into an active state. Since the MCU cannot do any useful processing until all of the digital and analog components are fully settled and operational, it is important to add in this (wasted) power when calculating total power consumption:

Total power consumed = Active mode power + Standby (sleep) mode power + Wake-up power


Power consumed during active mode
Power wasted during wake-up
Power consumed during sleep mode

Because every application is different, systems designers will have a tendency to weight some of these elements more than others. For example, some applications such as water meters spend most of their time in a standby state so clearly their long duty cycles require very low standby power consumption. Other applications such as data loggers go in and out of active states often so limiting the time spent in the wake-up transition modes is critical. However, a vendor developing a compelling MCU solution will not attempt to guess which of these metrics is most important but instead will design a solution from the ground-up that focuses on minimizing every part of this equation. To accomplish this requires strong mixed-signal expertise to address both the architectural-level and circuit-level challenges necessary to minimize power in both the analog and digital domains. A short discussion on each of these variables will help highlight the types of issues systems designers need to be aware of when attempting to select the best MCU solution for their application.

Active Mode Current

For a CMOS logic gate, the dynamic power consumption can be rewritten using the following well-known equation:

Active mode power = $C * V^2 * f$
where C is the load capacitance, V is the supply voltage, and f is the switching frequency

The capacitance term is a function of the design and processing technology being used, and the frequency term is a function of the application's processing requirements. However, as can be seen in the equation above, the supply voltage has a disproportionate impact on the overall power consumed by the MCU. Therefore, adding voltage regulation to the MCU design can yield significant active mode power savings by providing a much lower steady supply voltage to the MCU's circuitry. Switching-type converters may be a possible solution but they are best suited for regulator environments requiring large voltage conversion ratios. However, for battery-type applications where the average voltage conversion ratio is small (approaching 1:1 at the end of the battery life), a better solution would be to add an on-chip low drop-out (LDO) linear voltage regulator since it can offer acceptable efficiency with lower complexity and cost than a switching solution.

To illustrate the benefits of using an LDO regulator, it is helpful to restate the CMOS dynamic power equation:

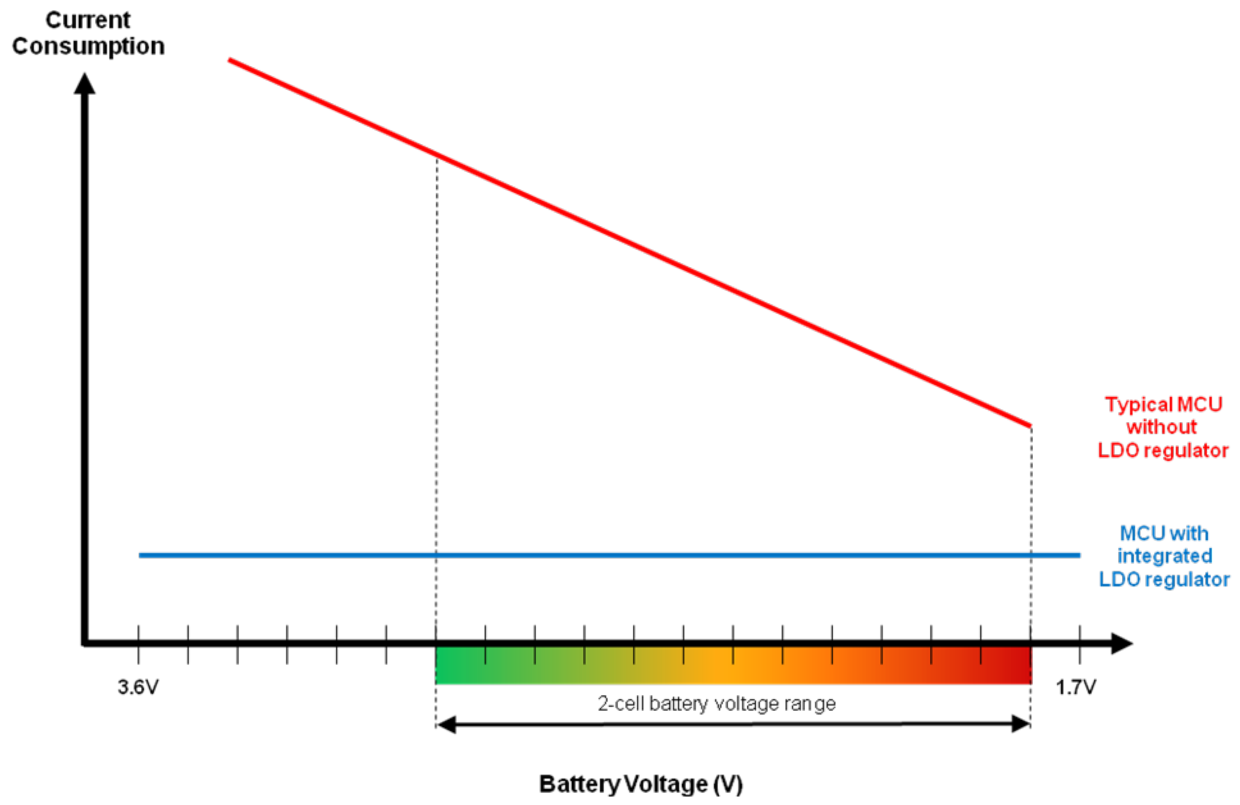
Active mode power = $C * V^2 * f$

$= V * (C * V * f)$

$= V * I$, where the dynamic current $I = C * V * f$

It is common to normalize the dynamic current to a frequency of 1 MHz and a particular supply voltage. For example, one recently introduced ultra low-power MCU has a dynamic current consumption of 160 μA per MHz at 1.8 V. Without supply regulation, this metric would increase to (160) * (3.2/1.8) = 284 μA per MHz when the supply voltage is 3.2 V. With an LDO, the battery current will remain fixed at 160 μA per MHz across the entire supply range.

As can be seen, this advanced power architecture can be used to maintain a constant active current over the full operating voltage range and can help systems designers achieve a significant savings in power consumption. Therefore, from a systems designer perspective, it is important to determine the MCU current consumption when operating across the entire operating voltage range - not just at the 1.8V minimum operating condition that is commonly quoted by MCU vendors. Quoting an overly optimistic current number that assumes anything less than a typical voltage supply does not accurately reflect how applications are used in the real-world. As an example, in 2 x AA/AAA and coin cell battery applications, the batteries operate near their 3V initial voltage most often so the quoted 1.8V specification can be deceiving since when viewed from this perspective most MCUs will consume around 50 percent more power than what is commonly quoted.



Furthermore, since power consumption is directly proportional to the switching frequency, it is important for systems designers to normalize the quoted current numbers down to a current / MHz basis. By combining these two factors, it is possible to perform a side-by-side comparison of MCUs based on the following metric:

- Current consumption / MHz @ 3V

Some vendors will attempt to confuse the issue by equating “MHz” to system clock speed when the value that is truly meaningful is instruction clock speed. This is deceiving since system clock speeds can run at twice (or more) their actual instruction speed thereby doubling (or more) their effective power consumption. It is therefore important to make sure everything is normalized to instruction clock speed. By doing this, and by using a typical supply voltage, it will be possible to properly derive the actual active mode current consumption budget.

Standby (Sleep) Current

Achieving maximum energy efficiency (and battery life) translates into ensuring that each MCU task consumes the minimum possible current at the minimum possible voltage for the shortest possible duration so that the device spends the majority of its time in a very low-power sleep mode. In some applications, the sleep-mode current is the parameter most responsible for overall energy consumption. However, what is often overlooked is that the absolute minimum sleep current achievable by an MCU is primarily limited by its leakage current. For example, a 20-input device that has an input leakage current specification of 100nA could consume up to 2uA of power while in sleep mode.

Leakage current is affected by a number of factors, but the most important one is the underlying process technology that is used. In some cases, vendors will choose to use 0.25 or 0.35-micron process technology to reduce the sleep current caused by leakage, but this choice comes at the expense of a higher active current. In other cases, MCU vendors choose to use 0.18-micron or smaller process technologies to reduce active mode current, but this comes at the expense of higher leakage currents. A unique solution around this dilemma is to apply mixed-signal expertise to implement an advanced power management unit (PMU) designed specifically to limit leakage and to enable ultra-low sleep current regardless of the underlying process technology that is used.

When using process technologies of 0.25-micron or smaller, minimizing sleep-mode current requires cutting power to the digital core. Modules that operate in sleep mode, such as power management circuits, I/O pad cells, and an RTC, must operate from the unregulated voltage supply to avoid burning additional current in an LDO. Cutting power to the digital core logic also prevents its off-state leakage from contributing to the sleep-mode current; however, the MCU must preserve RAM contents and the state of all registers during sleep mode so that code execution can resume right where it left off. This preservation may be performed either by means of some very low-current sleep-mode latch biasing scheme or by the use of special retention latches that can hold the state in sleep mode without significant leakage. The MCU also needs some form of continuous supply voltage monitoring (i.e. “brownout detection”) to reset the device in the event that the supply voltage drops below the minimum retention voltage, which could corrupt the state.

From a systems designer perspective, it is therefore important to examine the underlying leakage current specifications to determine which MCU vendors have applied their mixed-signal expertise towards solving this complex problem. Designers should also consider the fact that most vendors offer many different standby current options. Most suppliers will highlight their absolute lowest sleep mode current, which will often correspond to the current being consumed with the real-time clock and brownout detector disabled. Some vendors will go a step further and quote a shutdown mode current that does not retain memory and requires a reset to wake up, which in general is not a very practical mode. Therefore, since most applications will require full RAM and register retention, it is important for a system designer to perform a side-by-side comparisons based on the following metrics:

- Standby/sleep mode current with real-time clock and brownout disabled (with RAM retention)
- Standby/sleep mode current with real-time clock disabled and brownout enabled
- Standby/sleep mode current with real-time clock and brownout enabled

A system designer can then use the correct values when calculating the overall standby mode power budget based on the duty cycle of their application.

Wake-up Energy

As discussed earlier, in systems that use sleep modes a significant amount of power can be wasted waking up the MCU and preparing it to acquire or process data. In fact, in certain applications an MCU can often use just as much energy when coming out of standby as when the device is fully processing data. Therefore, it is important to design an MCU to wake-up and settle in an extremely short amount of time in order to minimize the amount of time spent in an energy-wasting state.

The MCU should be able to exit sleep mode from either an external trigger event or an internal timer. The most flexible periodic wakeup source is a real-time clock having the capability of being run from an external crystal oscillator (for applications requiring accurate timing) or from a low-frequency internal oscillator that eliminates the need for a crystal in lower-accuracy applications. Avoid using a slow-starting crystal oscillator for the high-speed system clock; an accurate, quick-starting, on-chip oscillator is a better alternative.

In addition, since many products wake up periodically to sample an input using on-chip ADCs, it is important to allow enough time for both the digital circuitry to wake-up and the analog circuitry to settle to begin making valid measurements. The startup behavior of the analog modules can have a major impact on the amount of time spent in active mode; voltage regulators or references utilizing external decoupling caps can take milliseconds to settle. At times, MCU vendors will only quote the wake-up times for the digital circuitry while ignoring the time it takes for the analog circuitry to settle. Therefore, it is important for a systems designer to analyze the overall wake-up and settling time for both the digital and analog circuitry to factor in the true cost of this wasted energy.

Other Considerations

There are of course other ways to further reduce power in a system. For example, 2 x AA/AAA battery configurations are commonly used due to the fact that MCUs often can typically only operate down to 1.8 V and even then, sometimes only with reduced functionality (no ADC; reduced instruction clock speed). An innovative way to reduce power (and environmental impact) is to convert the design to a single battery configuration where the battery can be operated all the way down to the end of their useful life (0.9V). To enable this, an MCU must integrate a highly optimized DC-DC converter that can be operate to the lowest usable voltage of the battery, which in the case of alkaline chemistry is 0.9V. This approach can also save the supplier and/or the consumer the cost of a battery.

Another method for reducing power is to use highly integrated MCUs that include ADCs, DACs and other peripherals since the MCU can be given control over enabling and disabling these peripherals as needed by the applications. For example, some MCUs offer a specialized low-power ADC with burst mode that can take analog measurements while the CPU is off in an effort to further minimize power consumption.

Summary

It is understandably a challenge to parse through the various conflicting claims presented by MCU vendors. However, for a majority of applications, it may be best to simply revert to the fundamental power consumption equation to cut through the clutter:

Total power consumed = Active mode power + Standby (sleep) mode power + Wake-up power

Because every application will be affected by the combination of standby power, active mode power and wake-up power, it may be helpful for systems designers to simply start any analysis by systematically breaking down the power consumption numbers into the parts shown above. Once these numbers have been derived, a system designer can then factor in the application’s duty cycles – the amount of time the application expects to spend in standby, active and wake-up modes – to calculate an overall average power consumption number. The resulting value should provide the system designer a close approximation that can be used to objectively evaluate and compare MCU alternatives to achieve the lowest possible system-level power consumption.