SILICON LABS

How to Pick the Right Microcontroller Based on Low-Power Specifications

Introduction

Choosing the right ultra-low-power microcontroller (MCU) for your next embedded design can be a confusing task when you compare claimed current consumption specifications in a myriad of data sheets provided by MCU vendors. In many cases, developers initially scan the first page of a data sheet as a reference point to gain basic information about an MCU, including peripherals, operating speed, package information, number of GPIOs and power characteristics. This approach works well to assess an MCU's overall functionality, but it is not particularly useful when trying to gauge low-power characteristics.

To get a broader view of an MCU's true low-power operation, developers must take into consideration current consumption, state retention, wake-up time, wake-up sources and peripherals that are capable of operating while in low-power mode. Developers must compare a common operating mode to gain a balanced, apples-to-apples comparison among competing low-power MCUs. It is also important to take into consideration any additional functionality or peripherals that can reduce total system power and available evaluation tools that can make an engineer's job easier.

Microcontroller vendors will usually list the lowest power achievable on the first page of the data sheet. Although the device may be capable of achieving the specification in the data sheet, the actual operating mode may not be practical and useful in a real-world application. Some of the non-advertised features of the lowest power mode may include a very slow wake time, no state or RAM retention, or a reduced operating voltage range.

To get around the variety of low-power specifications, developers must identify a common operating mode consisting of two sections: electrical specifications and low-power functionality.

Comparing Electrical Specifications of Microcontrollers

The electrical specifications are available in the data sheet, but determining which specifications are relevant may require some digging. Usually the electrical specifications are organized by vendor-specific power mode. This makes assessment slightly more difficult, as it requires knowledge and familiarity with the functionality of each power mode.

In general, it is beneficial to define a set of operating conditions and then map them to a power mode. For example, the developer might define the following set of operating conditions:

- Sleep mode current consumption with state and RAM retention
    - All other peripherals disabled
- Sleep mode current consumption with RTC running with state and RAM retention
    - RTC enabled and running all other peripherals disabled.
- Wake time
- Supply voltage range

Once the operating conditions are clearly defined, it should be easy to determine the applicable vendor-specific power mode.

Additional Low-Power Functionality

The second section, low-power functionality, is not as easy to locate in the vendor's documentation and may be spread across the data sheet and reference manual. Examples of low-power functionality include:

- Available wake sources
- How code resumes execution
- Peripherals capable of operating in sleep mode.

Once the common operating mode has been clearly defined, developers can begin to examine the documentation in more detail.

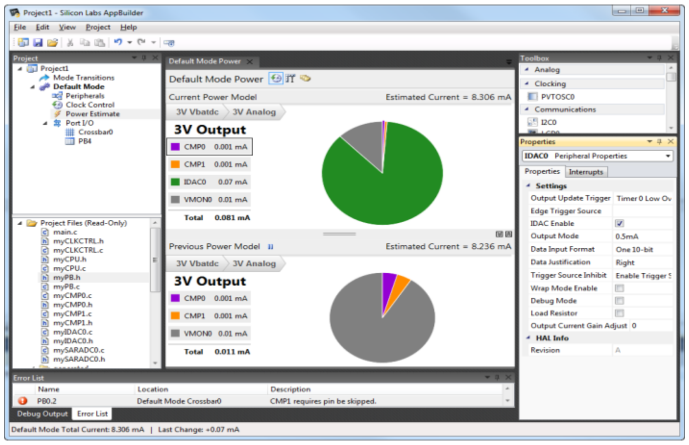
While going through this exercise of compiling data, keep in mind that there may be some MCU-specific features that can further optimize an application for ultra-low power. Optimizations may reduce bill of material (BOM) costs, provide longer product life or provide greater design flexibility. For example, an on-chip dc-dc converter can efficiently provide power to the system and decrease power consumption. This can enable the use of smaller batteries, which will decrease the overall BOM costs, or provide power budget flexibility. A variety of wake sources can provide design flexibility and allow the microcontroller to stay in the lowest power mode as long as possible, further reducing the average current consumption of the application.

Allowing firmware to scale the internal supply voltage is another optimization knob available to the developer. If an MCU is operating at a slow frequency, it may be possible to decrease the supply voltage and save power. Selective clock gating allows hardware blocks to be disconnected from the active circuits, preventing inactive peripherals from consuming power. These types of features are not comprehended by supply current specifications that are commonly used to rank low-power MCUs, but are critical to achieving the lowest overall system power consumption.

Reducing Complexity Using Tools

As MCUs become more and more configurable to achieve the lowest power consumption, they also can become more complex. To cope with this increased complexity, developers should take a close look at the evaluation platforms available for an MCU and the overall ease of implementing a solution. For example, the development board and software tools used to program the MCU should be intuitive and easy-to-use. Hardware that is difficult to understand or use is not likely to lead to an easy firmware development process. From a firmware perspective, MCU vendors should supply firmware examples that can recreate specifications from the data sheet. If advertised current consumption specifications cannot be recreated on an evaluation platform, it is likely that it will be just as difficult (if not impossible) to configure the MCU to achieve these numbers on custom hardware. Giving customers a variety of code examples that can be used as a starting point for their code development can reduce time-to-market and help engineers learn to use a device.

Graphical configuration tools can aid in development and help the developer gain a deeper understanding of an MCU. When developing low-power applications, it is helpful to know where the total consumed power is going. This information is useful because it highlights what aspect of a design needs to be further optimized and can also help the developer understand the overall architecture of the device. Ideally, low-power configuration tools could give tips on further reducing power as well as highlight any configuration errors that were detected throughout the configuration process. For example, the Power Estimator utility within Silicon Labs' AppBuilder graphical configuration tool provides Power Tips that give configuration guidance and a power-budget pie chart showing how much power is consumed and which peripherals are consuming the power. As configuration changes are made, the pie chart automatically updates.



Default Mode Power

Current Power Model — Estimated Current = 8.306 mA

3V Vbatdc › 3V Analog

3V Output

🟪	CMP0	0.001 mA
🟧	CMP1	0.001 mA
🟩	IDAC0	0.07 mA
⬜	VMON0	0.01 mA
	Total	**0.081 mA**

Previous Power Model — Estimated Current = 8.236 mA

3V Vbatdc › 3V Analog

3V Output

🟪	CMP0	0.001 mA
🟧	CMP1	0.001 mA
⬜	VMON0	0.01 mA
	Total	**0.011 mA**

IDAC0 Peripheral Properties

Settings

Output Update Trigger	Timer 0 Low Ov
Edge Trigger Source	
IDAC Enable	✓
Output Mode	0.5mA
Data Input Format	One 10-bit
Data Justification	Right
Trigger Source Inhibit	Enable Trigger S
Wrap Mode Enable	☐
Debug Mode	☐
Load Resistor	☐
Output Current Gain Adjust	0

HAL Info

Revision	A

Error List

Name	Location	Description
PB0.2	Default Mode Crossbar0	CMP1 requires pin be skipped.

Default Mode Total Current: 8.306 mA | Last Change: +0.07 mA

Figure 1. Power Estimator Enables Developers to Optimize for Lowest Current Consumption

To facilitate the microcontroller comparison process, the following table provides a list of common operating modes, as well as system-level optimizations and development tools available for Silicon Labs' 32-bit SiM3L1xx MCUs based on the ARM® Cortex™-M3 core.

MCU Specification	SiM3L1xx MCU Example	MCU2	MCU3
Sleep mode current consumption with state and 32kB RAM retention (All other peripherals disabled)	75 nA	-	-
Sleep mode current consumption with RTC running with state and 32 kB RAM retention (RTC enabled and running all other peripherals disabled.)	180 nA	-	-
Wake time	3.8 us	-	-
Supply voltage range	1.8 – 3.8 V	-	-
Available wake sources	Pin, RTC, comparator, pulse counter, LCD, low power timer, UART, charge pump	-	-
How code resumes execution (next line/reset)	Next line	-	-
Peripherals capable of operating in sleep mode	RTC, comparator, pulse counter, LCD, low power timer, UART,	-	-
System-level optimizations	On-chip DC-DC converter, static and dynamic voltage scaling, selective clock gating		
Software tools	IDE, AppBuilder graphical configuration tool, Power Estimator		

Summary

Evaluating and selecting a microcontroller for a low-power application requires more than a quick scan of the first page of the data sheet. Determining which MCU provides the lowest overall system power requires developers to know the device's supply current specifications, as well as any system-level optimizations that can reduce the overall supply current.

Unfortunately, each MCU vendor specifies operating conditions differently and in some cases advertises a low-power number that is available in an unusable mode. Using a common operating mode to compare MCUs will prevent developers from being misled by vendor claims of ultra-low-power operation.

Once the electrical characteristics of a device are understood and quantified, developers should take a look at the evaluation platform and software tools available. These considerations are crucial in getting an engineering team up and running quickly and should be included in the final microcontroller selection process. Find out more about Silicon Labs' microcontrollers, including 8-bit and 32-bit MCUs at www.silabs.com/mcu.

# # #