USB Connectivity in a Complex World

Designers of industrial and consumer devices have rapidly adopted the universal serial bus (USB) as the interface of choice for enabling connectivity to other applications due to its ease-of-use, plug-and-play functionality and robustness. Although USB connectivity has become a key requirement for most embedded applications, in most cases, it is just one of many design requirements for a typical application. For example, a blood pressure monitor that requires USB connectivity to enable the end user to download information to a PC must also perform its primary blood pressure measurement functions. This requires a complex set of interactions between pressure sensors and analog-to-digital converters for rapid data acquisition, intensive data manipulations to calculate blood pressure, and user interface design to properly display the results in a human-readable format.

In most applications, there is the additional requirement of achieving specific product cost targets, which adds yet another layer of complexity to already challenging designs. A highly-integrated USB solution not only enables the easiest path to achieving USB connectivity but can also provide the performance and analog capabilities required to enable developers to achieve their design goals in a cost-effective way.

A New Generation of USB MCUs

The rapid adoption of USB in industrial and consumer applications is challenging embedded solution designers to incorporate USB connectivity into their products while maintaining or, in some cases, reducing overall costs. Early versions of USB-based microcontrollers (MCUs) were developed to enable the addition of USB, but they lacked the capability to support other functions or peripherals. In the early days of USB, these devices played a key role in boosting the overall popularity of the USB interface. Even today, these bridge devices can be effective solutions that quickly enable the addition of full-speed USB via a companion chip, thus avoiding the need to redesign entire systems. However, for cost-sensitive applications, this approach may not be ideal.

To overcome this cost penalty, the new generation of USB-based MCUs incorporates a greater number of functions and peripherals. However, although the number of USB-based MCUs with different combinations of peripherals has grown significantly, a gap still exists for highly-integrated solutions that incorporate not only the right mix of peripherals but also ensure that these functions are sufficiently robust to support critical application requirements. Although application requirements are highly-dependent on the characteristics and functionality of the final product, there are three main areas of interest common to most applications.

The first area that will be explored is CPU performance and the impact of USB when it is included as an integrated peripheral. The second area to consider is the analog functions or peripherals that play an important role when interfacing with real-world signals. Finally, since almost every application is cost-sensitive, a USB implementation that reduces cost by eliminating external components is highly desirable.

Most often, the CPU is responsible for executing user code, and its ability to execute instructions and process data in a timely manner is paramount. A typical cost-effective USB controller incorporates a first-in/first-out (FIFO) function block to manage the incoming and outgoing USB packets while the CPU is used to read and write data to/from these buffers, in addition to performing other tasks.

Interplay between the CPU and USB Function

Let's examine a USB-to-serial bridge application to describe how CPU performance can be impacted. In this bridge application example, assume that the requirement is to bridge a serial-based UART device with a USB-based system. At the simplest level, the CPU should be capable of taking data from the UART interface (UART FIFO) and placing it into the USB FIFO, and vice versa. However, what if this same application needs to perform other simple functions, such as reversing the endian ordering, or complex functions, such as applying software filters? What began as a simple task suddenly becomes a much more complex operation that requires special attention to properly manage and places an increasingly large burden on the CPU.

A typical protocol bridge is expected to pass data from one peripheral to another, practically in real-time; so, the CPU needs to have the performance necessary to read, write and manipulate data with an acceptable latency. Silicon Labs' C8051F38x and C8051T62x USB MCU families are good examples of low-cost USB solutions with an enhanced high-speed 8051 CPU core that can execute 70 percent of its instructions in one or two system clock cycles. This type of performance is sufficient to not only meet the needs of protocol bridge applications but also to address most other full-speed USB applications. As an additional benefit, a high-speed CPU executes more work in less time, which can, in turn, reduce overall power consumption by enabling a system to stay in low-power mode longer.

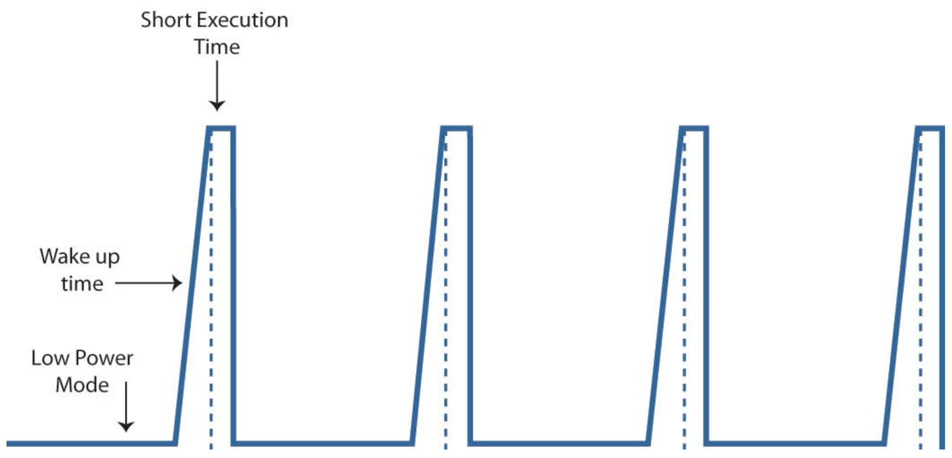

Figure 1. Energy-Efficient MCUs Enable Systems to Stay in Low-Power Mode Longer

Incorporating Analog Functions in USB Solutions

Analog functions or peripherals, such as analog-to-digital converters (ADCs) and comparators, are commonly used across many applications. For example, ADCs and comparators are used in everything from basic battery management implementations to highly sophisticated data acquisition systems in high-speed sensor interfaces. To support such a wide range of applications, ADCs and comparators must be robust and possess enough features to address these varying requirements while still being sufficiently inexpensive to be integrated into an USB MCU.

For example, many of Silicon Labs' USB MCUs feature on-chip, high-performance optimized ADCs. These ADCs feature a 500 ksps conversion time with a track and hold capability that enables the insertion of clock cycles after each ADC conversion. Specifically, each conversion is preceded by a tracking period of three ADC clock cycles after the start-conversion signal. This mode is very useful when multiple ADC channels are in operation because it enables the proper settling time necessary for an accurate conversion. Additionally, a programmable window detection feature can be used to compare the ADC output registers with user-programmed limits. This feature is especially desirable in battery management applications in which the user sets a limit on how low the battery can go before an alarm is triggered. In addition, because no CPU intervention is required to implement this feature, there is a very short latency period, which further enhances the safety of these types of battery applications.

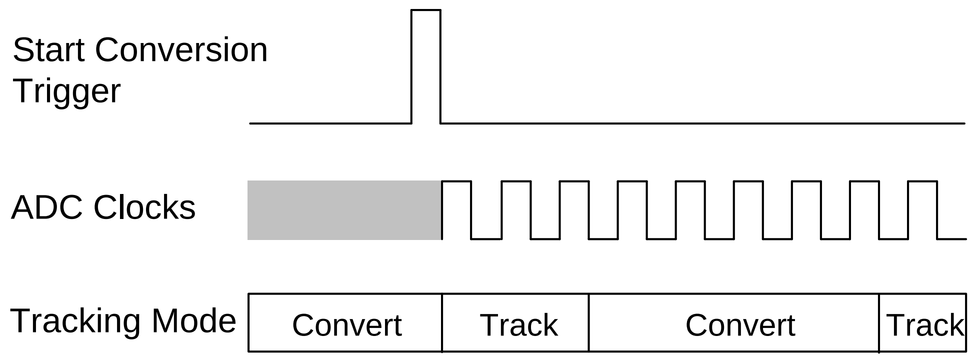

Figure 2. ADC Tracking Mode

Comparators provide another very useful analog function widely used in many applications. Typical implementations can be found in devices, such as blood glucose meters, in which a comparator is used to detect the insertion of a test strip, or insulin pumps, which require a fast shutdown mechanism in case the motor pump stalls. In these two examples, among many others, the response time and power consumption of the comparator are critical. While traditional USB-based devices have a loose specification for these comparators, some members of the C8051F38x and C8051T62x/32x families provide comparators with programmable response times as low as 100 ns. Power consumption is also user-selectable and can be as low as 1 µA. When viewed in this context, it is possible to achieve analog performance numbers in MCU-based devices comparable to those found in typical standalone analog ICs. USB-based MCUs with integrated high-performance analog features can provide a cost-effective, single-chip solution by replacing external analog chips.

Benefits of Integrated USB MCU Solutions

The integration of USB into a single-chip MCU solution may require a different way of thinking about how best to cost-optimize the system design. For example, adding USB to a design may have a substantial impact on the way the clock tree system is designed. To ensure reliable USB connectivity, it is critical that USB clock accuracy be maintained. Typical USB-based MCUs require designers to add an external crystal and associated components to meet USB clock accuracy requirements. This approach not only increases the cost of the solution but also increases the PCB design complexity and overall size. In addition, external termination resistors are often required to identify the USB speed, further increasing the cost of the USB implementation.

The clock recovery capability integrated in the C8051F38x and C8051T63x/32x USB full-speed devices is an excellent example of an innovative feature that eliminates the external crystal commonly required on other USB-based microcontrollers by enabling the internal oscillator to adjust itself based on the incoming USB data stream. This allows the internal oscillator to meet the requirements for USB clock tolerance. In addition to the cost reduction achieved by removing external components, the elimination of the external crystal also brings another major benefit. Electromagnetic interference (EMI) is dramatically reduced by eliminating the clock-related noise emissions. Additionally, these solutions have integrated termination resistors that are fully software-controllable. The elimination of the external crystal and its associated components and the integration of termination resistors are major steps forward in helping designers reduce the cost and complexity of adding USB to a design.

Another challenge commonly associated with USB design is the need for complex and time-consuming software development. The availability of USB drivers and code examples can help overcome this hurdle and dramatically shorten development time. For example, Silicon Labs' USB MCU development kits include production-ready host and device USB drivers. No USB protocol or host-device driver expertise is required to use these kits, which enables a quick and easy way to implement USB connectivity.

Summary

USB connectivity is a key requirement for many embedded applications. Highly-integrated USB MCU solutions not only offer the easiest path to achieving USB connectivity but also provide high-performance CPU cores coupled with integrated analog capabilities that help reduce component counts and BOM costs. USB MCU solutions enable embedded system developers to dramatically simplify their designs while reducing costs.