Whitepaper 4: Maximise Your Battery Life

Abstract

Battery powered or battery backed designs are becoming more common, but with the ever growing need for greater efficiency and use of resources (i.e. batteries), designers are coming under increasing pressure to make the battery last longer, if not for the life of the product. Many of these applications operate on a cyclic basis with the system spending a large amount of its time asleep, where very low average current especially during standby play a critical part.

Introduction

This paper is the last in the series of four whitepapers examining the various techniques necessary for low power design and operation in battery powered applications. In this paper we examine the criteria for battery powered systems that spend large amounts of their time asleep (standby), where the goal is to make the life of the battery last as long as possible.

Much of the detail has already been provided in the previous three whitepapers, so to avoid duplication this paper summarises these key areas that should be taken into consideration. References used are based on the Renesas 16-bit RL78 and the 32-bit RX100 families as outlined in the previous papers.

Battery operated systems that spend most of their time asleep

Whitepapers 1, 2 and 3 examined much of the detail surrounding the different sleep (standby) operation and the effects of combining low system clock frequencies with the use of standby. For long battery lifetimes it is essential that standby modes are used during system idle times as this will significantly lower the average power consumption in all applications (refer to white paper 3).

However as outlined in whitepaper 2, standby can have its own problems such as

1. Stopping peripherals and resetting their state and configuration settings
2. Analogue functions remaining powered or having slow initialisation times
3. Managing external peripherals

Ideally all peripherals should be turned "off" during standby to provide the lowest possible current drain, but as this can cause issues it is important that the effects are analysed to determine the best overall solution and managed as necessary within the system software.

Summarised below are the key criteria with details and references taken from the other three whitepapers that affect applications requiring long standby operation.

Standby

Paper 1 gave a simple example of a repetitive (cyclic) application where operating the MCU at a low clock frequency (32 KHz) combined with using standby operation, long operating times and battery life (or in this case the long life of a lemon) were achieved.

Battery life is improved further in applications where the system can be "stopped" (i.e. all clocks stopped) during the idle periods so that all activity is frozen until a wake up trigger is generated. This is shown in whitepaper 3 which examines what impact the combination of low clock frequencies and using the different standby modes has on the average currents results.

Taking some examples from the other papers very low average currents can be achieved, based on the equation below:

$$(I_{ave} = ((AC1 * AT1) + (AC2 * AT2) + (IC * IT)) / P)$$

We saw average currents of 400 nA for the RL78/L12, 800 nA for the RX111 and average levels of 2 µA to 3 µA when peripherals such as the LCD display is operating (i.e. RL78/L12).

Peripherals

Generally most internal or external peripherals are not required to operate during idle periods and should ideally be "stopped" to avoiding any unnecessary power drain. However be sure to check the MCU user manual in order to properly understand the operation of all the peripherals during standby and the effects of stopping (resetting) and re-starting any peripheral used in this way, as it is likely that additional code and time may be needed to re-initialise the peripheral.

The current drain of analogue functions (both internal and external) is not solely related to clock operation but due to "static" currents (references resistor ladders etc.) present with all analogue functions. The optimum approach for using analogue functions is to initialise, use and then stop (turn off). However please be aware that they require relatively long initialisation times so this time needs to be allowed for in the system. It is an ideal time for the CPU to perform other foreground tasks while waiting for the analogue function to be ready for operation.

I/O Pins

Unused I/O pins should not ideally be set as inputs (often the default state) and left unconnected (i.e. floating). Rather having to use internal (or external) resistors these pins could be set as outputs so pull up/down resistors are not required.

Where resistors have to be used (pull up/down or functional) look to use the minimum number required and the highest value possible to maintain correct operation of the function. Internal resistors are pin programmable by software, so that only those needed can used with all the others turned off. Specify the highest value possible as, for example, internal pull up resistors can have a value as high as 100 kΩ, so any external can be also be this value, whereas open drain (typically used for the I^2C interface) will need to be considerably lower (can as low as 1 KΩ) to maintain correct rise and fall times to the specifications. Refer to the user manual for the MCU device used.

Power Supply

Generally use the lowest voltage that can be supported by the complete system and while this will not make any difference to the current used by the MCU when operating at 3 volts or 5 volts (and in some cases as low as 1.8 volts), it can have an impact the power used by the system outside the MCU.

As many MCU's can operate down to 1.8 V (or even lower) this provides the possibility to operate over a wider voltage range extending the battery pack operating range and life.

It may be possible to reduce the power supply during standby times where the minimum voltage is used to maintain register settings and RAM contents. However this can be quite a complicated procedure and care should be taken to sequence the "power down" and "power up" of the system so as not to affect any external peripheral connected to the MCU. This is also likely to extend the time taken for the "power down" and "wake up" of the system.

System Integration

As outlined in whitepaper 2, external digital peripherals require high current I/O pins to interface to the device. Using MCU devices which include these functions on-chip can significantly reduce current consumption as the internal interfaces require much lower currents due to lower voltages and capacitances (½ C * V2 * f) and these peripheral functions can easily be disabled when not required.

This can also apply to analogue peripheral functions where modern MCU's are increasingly including analogue functions (comparators, PGA, references etc.) in addition to the conventional ADC and DAC functions, although a need for very high accuracy or performance may dictate the need for external analogue devices. It is important here that the design can either turn the device off or switch off the power supply to these devices during standby operation.

Conclusion

While the theme of this paper was to look at the criteria for battery powered systems spending large amounts of their time asleep (in standby) and as much of the detail and examples are provided by the previous three whitepapers, this paper purely highlights and summarises the key areas affecting operational and battery life but without duplicating all the details. Also as this is last in the series of (four) papers and the topics throughout are linked, the conclusions include those from the previous papers to provide a final view of low power management.

Applications demanding low active and standby current consumption especially those spending the much of their time asleep (i.e. standby) should follow a thorough design and test process (i.e. a "Power Use Profile") to compare the design goals against the actual consumption and highlight areas requiring rework in order to achieve the lowest average current possible and meet the design goals for the application.

The topics highlighted in this paper together with the "tricks" outlined in whitepaper 2 help to define how to manage the power consumption with whitepaper 3 showing that just operating with a reduced continuous fixed clock frequency provides little usable power reduction. By operating with a dual clock scheme of a sub-clock during the standby times and a higher frequency clock during the foreground periods (but not using standby modes), did show some good improvements in power use which could possibly be used as a solution for some applications.

However it is clear that using the MCU standby modes during idle periods provides the lowest average current, confirming that this is the best design approach wherever possible to make the battery last the longest.

In our examples for the series of papers two MCU's families were used, a low power 16-bit MCU (RL78/L12) and higher performance more integrated 32-bit MCU (RX111). The question was asked whether a 32-bit MCU providing increased functionality and performance can really still offer low power. The answer is yes, that when employing all of the techniques outlined in these papers, a modern 32-bit MCU family can produce very low average currents while delivering higher performance and functionality when the application demands are higher.

In order to understand the full background and details behind the above conclusions, it is recommended to read the preceding three whitepapers in this series (highlighted below for reference) and visit the Renesas design resources centre.

Whitepaper 1: Lemon Powered Design

An example of what can be achieved with the right product and modes of operation

Whitepaper 2: The Rules of Low Power MCU Design

Analysis of many techniques to provide the lowest power consumption possible

Whitepaper 3: De-Clocking vs MCU Standby for Low Power Design

Reducing MCU clock speeds during operating and idle times and effects of combining with standby

Written by: David Parsons - Consultant to Renesas Electronics (Europe) GmbH
David can be contacted at DCP Electronics and Software Services.

Document No. R01PF0081ED0100