CYMBET™ CORPORATION

AN-1063
Application Note

Extending Battery Life Using The CBC921 PMRTC

Introduction

This Application Note discusses techniques to use the CBC921 Power Management / Real-Time Clock (PMRTC) to reduce system power in microcontroller-based applications. Such techniques extend battery life significantly, resulting in longer device operating time with a battery of a given size and giving product designers the option to use smaller, often less expensive, batteries that can operate for months and years - improving system dependability and reducing replacement and disposable costs.

CBC921 PMRTC in 3mm x 3mm x 0.5mm 16-pin MLPQ Package

Power Reduction Techniques

Typical microcontroller-based devices can benefit from the power saving features in many microcontrollers. These features include ways to shut down portions of the chip such as serial ports and other peripherals when they are not being used, changing clock frequencies on the fly, or going into a variety of “sleep” modes that can drastically reduce power by powering down subsystems not currently in use. The most common and generally most effective method is to put the microcontroller and peripherals into sleep with only a wake-up timer running. This mode can often reduce the current of the microcontroller to a few microamps or less of current. Even so, the sleep current integrated over a long time can be a significant power drain. Some microcontrollers can stop all timers and operations but keep a few registers alive to cut the current to only several tens of nanoamps but they don’t have a built-in way to wake up since the internal timers are also suspended. The microcontroller power reduction through sleep is a big benefit but often the sensors or user controls must stay awake which increases the power usage.

This Application Note describes methods of using a CBC921 PMRTC to utilize the lowest power suspend modes of microcontrollers and/or to completely shut power down to peripheral chips for periods of time to **drastically reduce power in systems where low power consumption is essential,** such as remote sensors, timers, data-loggers, and battery-powered devices generally. The techniques and results will be discussed first, followed by considerations for systems with higher active power. Lastly, specific registers that are used to set up the CBC921 to accomplish such power savings are listed. **System power savings of factors of ten or even one hundred are possible.**

Sleep Power versus Active Power

Typical microcontroller-based devices where energy conservation is an issue spend a large part of their time in a low-power “sleep” state. This is a state where the microcontroller is not running but is waiting for an interrupt from either a sensor or a timer. When one of these interrupts is issued the microcontroller goes to a higher power active mode to process the event and then goes back to sleep. The active mode operation may include processing the sensor data and then may operate an actuator or send a message. Many times a message may be sent via a low-power radio protocol which requires significant processing cycles to correctly operate the protocol stack. The amount of processing depends greatly on the complexity of the protocol.

The average power consumption of the system is the sleep power times the percentage of time the system is asleep plus the active power times the percentage of time the system is active divided by 100.

$$P_{avg} = (P_{sleep} * \%\text{ time asleep} + P_{active} * \%\text{ time active})/100$$

Minimizing the largest of these terms provides the greatest power savings. In some cases the active power term is much larger than the sleep power term either because the power per event is large or the active power events happen very often.

Average Power Consumption when Mostly in Sleep State

When the system that has a large sleep power compared to its active power is asleep, there is an opportunity to reduce power by placing the microcontroller in its lowest power mode while using the CBC921 RTC timer functions to provide periodic wake-ups to the microcontroller and associated circuitry. This way the entire system is totally asleep for the majority of time and the microcontroller is only awakened for short periods to determine if it needs to service a sensor or switch. The CBC921 is configured to automatically wake the system at regular periods for a finite time and then the microcontroller goes back to sleep. The CBC921 can also be configured to completely turn off power to the sensors and/or microcontroller by using its internal 4Ω pull-down switch. If the microcontroller determines during one of its waking intervals that it needs to service something then it quickly commands the CBC921 to not automatically shut it down until further commanded. This technique can reduce the system power greatly, as the CBC921 only requires 15-50nA of current to manage the timing functions and the rest of the system can go to its lowest power mode.

The average system power is a function of the time the system needs to run compared to the time it is asleep. If the microcontroller/system is awakened too often the power savings will be minimal. A simple metric of the possible savings is to add the sleep current (since it is always present) to the active current times the ratio of the active time divided by the sleep time. Table 1 below shows some examples of different power savings that can be achieved with different sleep vs. run times. Column one is the Original Sleep Current of the system without using the CBC921. The Original Sleep Current includes the sleep power of the microcontroller with a timer running plus any sensor current. The Power Savings Ratio is the Active Current times the ratio of active/sleep times plus the 15-50nA CBC921 current compared to the Original Sleep Current in column one. The Number of Instructions column shows how many instructions the microcontroller can execute in the period of time listed in the Active Runtime column. For sake of reference it takes about 28 I2C clocks at 400kHz or about 70μs to write to a single register in the CBC921. Make sure to write to the register to disable the timer before the CBC921 automatically switches the microcontroller/system power off.

Table 1: Combining Sleep Power and Active Power to Compare Power Savings.

Power Savings with Periodic Interrupt								
Original Sleep current (μA)	Active Current (μA/Mhz)	Clock Speed (Mhz)	Active Runtime (ms)	Sleep Period (ms)	Active Current When Running (μA)	Active Time / Sleep Time	Number of Instructions	Power Savings Ratio
0.6	200	1	0.1	16	200	0.0063	100	0.47
0.6	200	1	0.1	33	200	0.0030	100	0.97
0.6	200	1	0.05	33	200	0.0015	50	1.89
0.6	200	1	0.1	100	200	0.0010	100	2.80
0.6	200	1	0.05	100	200	0.0005	50	5.26
0.6	200	1	0.1	250	200	0.0004	100	6.38
0.6	200	1	0.05	250	200	0.0002	50	11.11
0.6	200	1	0.2	250	200	0.0008	200	3.45
1.6	200	1	0.3	250	200	0.0012	300	6.30
1.6	200	1	0.5	1000	200	0.0005	500	14.04

Notice that the Power Savings Ratio is only a benefit if it is over 1.0. This table shows that the longer sleep periods have the best ratios. With higher Original Sleep Currents the benefits are also magnified. The next to the last line shows a Power Savings Ratio of 6.30 with over 300 instructions executed per wake-up. The system in this example had an Original Sleep Current of 1.6μA for the microcontroller current with internal sleep timer and an external sensor. **A 6.30 Power Savings Ratio means 6.3 times more battery life in a battery-powered system** (excluding battery self-discharge). The last line shows a one-second sleep time associated with a slower, environmental sensor. Note the 14.04 Power Savings Ratio in this example. These examples show that substantial battery life extension can be readily achieved using this technique.

Radio Power and Software Stack Concerns

The power savings technique described in the previous section works well with systems where the sleep power is a large or significant contributor to the average power. In systems where a radio is used, the power for the radio and the power used by the microcontroller to run the software stack can be large. Many radio standards are complicated and use a sophisticated stack to manage the protocol. These stack implementations often have significant runtimes to initialize their internal memory structures. Often these stacks are supplied by the radio chip vendor and therefore are attractive since they save software development time. If the stack needs a long time to initialize, then the previous power savings technique only makes sense if the radio operates very infrequently.

A typical radio application requires 25mA for 25ms, or 625μA-seconds of charge, for each transmission. A simple stack may take essentially no time to initialize but a sophisticated one may take upwards of 400ms of processing time at 5mA, resulting in a drain of 2000μA-seconds to initialize. Table 2 shows the effect of the radio current and the stack initialization current for this example at various reporting intervals.

Table 2: Effect of Active Power with Various Reporting Intervals.

Average Current vs. Transmission Interval				
Period Between Transmissions	Average Sleep Current (μA)	Average Transmit Current (μA)	Average Active Current for Stack Initialization (μA)	Overall Average Current (μA)
1 Second	1	625.0	0	626.0
10 Seconds	1	62.5	0	63.5
1 Minute	1	11.0	0	12.0
10 Minutes	1	1.1	0	2.1
60 Minutes	1	0.2	0	1.2
1 Second	1	625.0	2000.0	2626.0
10 Seconds	1	62.5	200.0	263.5
1 Minute	1	11.0	33.3	45.3
10 Minutes	1	1.1	3.3	5.4
60 Minutes	1	0.2	0.6	1.7

Design Technique: Notice the effects on overall average current both with transmission period and with stack initialization. This highlights the importance of implementation: use a simple stack initialization, or a microcontroller that has a low power mode to retain the initialization in RAM at low power, or both.

Example System Configurations

The system can be configured to cut power to the sensors and microcontroller as shown in Figure 1 and Figure 2. The additional FET in Figure 2 allows the VCC bus to be switched instead of VSS. The PSW/nIRQ2 output includes a selectable 1Ω FET switch that greatly reduces drops in the switched bus.

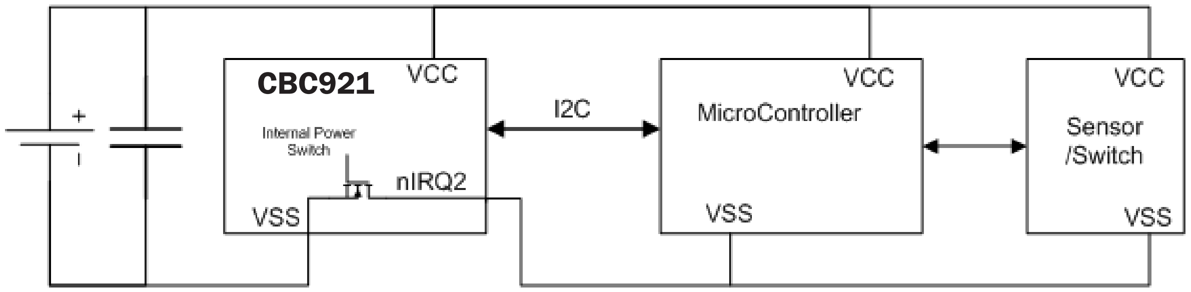

Figure 1: Switched Ground Configuration.

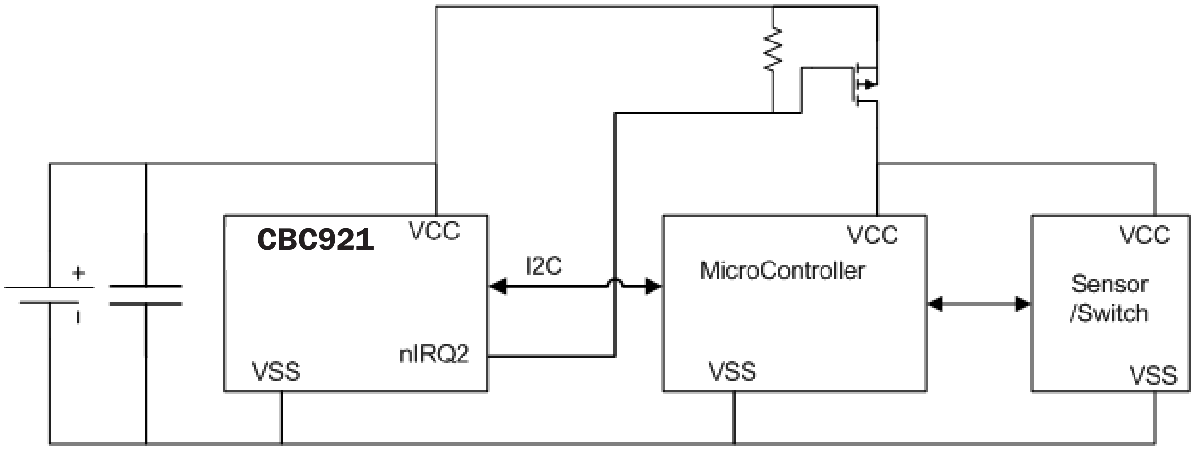

Figure 2: Switched VCC Configuration.

It is also possible to not switch the power bus, but instead put the microcontroller in deep sleep and wake it periodically with an interrupt as shown in Figure 3. This works well with microcontrollers that have higher internal sleep timer currents but very low current when clocks are disabled and only some memory is retained.

Design Technique: Firmware must be designed to avoid large memory re-initialization routines on startup either from reset or from the interrupt.

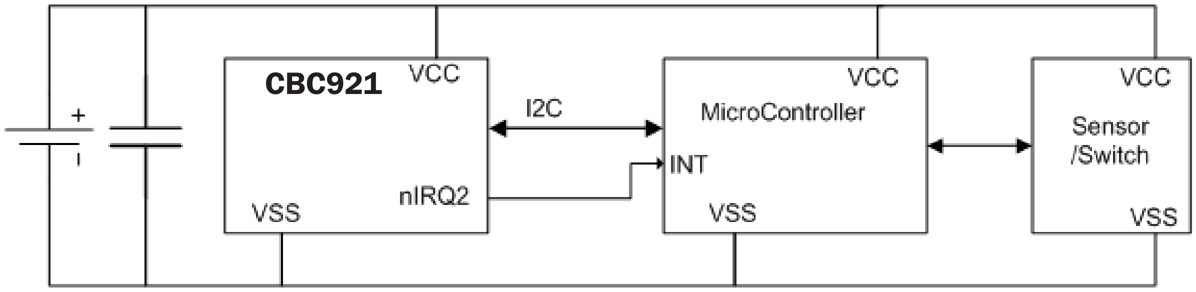

Figure 3: Interrupt-Only Configuration.

Important CBC921 Registers to Consider

Control Register: OUTB -> This is a static default value which can be driven on the PSW/nIRQ2 pin, depending on the OUT2S and ENPSW_WAKE settings. If the OUTB value is selected by OUT2S and enabled by ENPSW_WAKE, a value of 0 pulls the PSW/nIRQ2 pin low using the internal low-resistance switch; whereas a value of 1 leaves this pin open drain, which could disconnect power to external components if the PSW/nIRQ2 pin is being used as a power switch.

Control Register: OUT1S -> Selects the function of the nPGD/nIRQ pin.

Control Register: OUT2S -> Selects the function of the nIRQ2 pin based on Table 1, assuming both logic states are allowed by ENPSW_WAKE when not in SLEEP.

Timer Management Register: Countdown Timer Enable CDTE -> A value of 1 enables decrementing in the Countdown Timer, as reported in the Countdown Timer register. Disabling the timer with a value of 0 stops the countdown but retains the latest value in the Countdown Timer register.

Pseudo-Code Examples

Demonstration code examples can be found on the Cymbet web site: https://www.cymbet.com/resources/documents-and-downloads/

Ordering Information

Part Number	Description	Notes
CBC92141C-Q3	Power and Battery Management / Real-Time Clock I2C 4.1V Vchg 3x3x0.55 MLPQ-UT16 Package - Cut Tape	Digi-Key P/N: 859-CBC92141C-Q3-TR3CT-ND
CBC92141C-Q3-TR3	Power and Battery Management / Real-Time Clock I2C 4.1V Vchg 3x3x0.55 MLPQ-UT16 Package 3k Reel	Digi-Key P/N: 859-CBC92141C-Q3-TR3-ND
CBC92132C-Q3	Power and Battery Management / Real-Time Clock I2C 3.2V Vchg 3x3x0.55 MLPQ-UT16 Package - Cut Tape	Digi-Key P/N: 859-CBC92132C-Q3-TR3CT-ND
CBC92132C-Q3-TR3	Power and Battery Management / Real-Time Clock I2C 3.2V Vchg 3x3x0.55 MLPQ-UT16 Package 3k Reel	Digi-Key P/N: 859-CBC92132C-Q3-TR3-ND
CBC-EVAL-14-PMRTC-32	ENERCHIP PMRTC EVAL KIT 3.2V CHG	Digi-Key P/N: 859-CBC-EVAL-14-PMRTC-32-ND
CBC-EVAL-14-PMRTC-41	ENERCHIP PMRTC EVAL KIT 4.1V CHG	Digi-Key P/N: 859-CBC-EVAL-14-PMRTC-41-ND

U.S. Patent No. 8,144,508. Additional U.S. and Foreign Patents Pending.