

# Low Power Design and Battery Operation with the MC1322x

## 1 Introduction

The MC1322x family is Freescale's third-generation ZigBee platform incorporating a complete, low power, 2.4 GHz radio frequency transceiver, 32-bit ARM7 core based MCU, hardware acceleration for both the IEEE 802.15.4 MAC and AES security, and a full set of MCU peripherals into a 99-pin LGA Platform-in-Package (PiP).

The MC1322x solution can be used for wireless applications ranging from simple proprietary point-to-point connectivity to complete ZigBee mesh networking utilized in automotive, portable monitoring devices and wireless dongles. Typically these applications need reliable usage throughout the day giving the developer the added complexity of proper power management techniques. Battery operation is key for many of the targeted applications. This implies optimizing low power operation in all functional operation modes: hibernate, doze, run, and idle modes.

### Contents

1	Introduction	1
2	Overview of Design for Low Power Operation	2
3	Configuring MC1322x GPIO for Best Practice	6
4	Minimizing Run Time Current	12
5	Summary Flow for Entering Low Power Mode	14
6	Managing Low Power from Freescale Software Stacks	14
7	Sample Application Demonstrating Low Power	15
8	Summary	19
Appendix A		
	Sample Application Demonstrating Low Power	20

## Overview of Design for Low Power Operation

In addition to being battery powered and portable, it is common for these devices to have the ability of wireless connectivity and seamless integration with other devices, portable or otherwise. The embedded designer has the responsibility of making these new products fit consumer requirements by creating applications taking full advantage of sophisticated power management and low power modes available in today's microcontrollers. More than ever, the developer must understand basic power management performance capabilities and wireless technology as consumers continue to require longer battery life and interoperability within their devices.

This application note describes requirements and techniques for using the MC1322x PiP in low power modes. It also describes how to use Freescale's embedded software together with practical use cases to enable low power management control implementation. By utilizing the different low power modes of the MC1322x transceiver, developers can optimize and increase application battery life eliminating the need to either charge or replace prematurely.

Following the initial hardware discussion and areas of considerations for low power applications, the use of Freescale's software products and tools are discussed illustrating how these tools can streamline writing of a low power oriented application. The software example targets SMAC but the BeeStack Consumer codebase (RF4CE) can also be used because the MC1322x is commonly used here.

The Freescale embedded software referenced in this application note is part of the standard BeeKit software development environment and can be found at [www.freescale.com/zigbee](http://www.freescale.com/zigbee) under Software and Tools, BeeKit Wireless Connectivity Toolkit under Current Updates and Releases.

### NOTE

It is recommended that the reader be familiar with the MC1322x device and is directed to the following documents for reference:

- MC1322x Data Sheet (MC1322x.pdf)
- MC1322x Reference Manual (MC1322xRM.pdf)
- 1322x-LPN Low Power Node Reference Manual (MC1322xLPNRM.pdf)
- BeeKit Wireless Connectivity Toolkit

## 2 Overview of Design for Low Power Operation

The MC1322x is inherently a low power device but also has extensive power management and an onboard buck regulation option to maximize battery life. The 1322x-LPN (Low Power Node board) reference design is intended to develop low power operation and applications code. It has been designed to allow low current measurement and support MC1322x low power application needs as referenced in this application note.

### NOTE

Refer to the *1322x-LPN Reference Manual* for additional board guidance.

There are two basic low power modes of Hibernate and Doze, and both have options of how much RAM contents are retained. The primary difference between Hibernate and Doze is Doze mode keeps the

primary reference oscillator running. The following low power modes must be considered for absolutely lowest energy usage, i.e., longest battery life for your application type:

- Low power modes
  - Off (reset active)
  - Hibernate (capable of operating current from <math><1\mu\text{A}</math> to about 5 $\mu\text{A}</math>)$
  - Doze (capable of operating current of <math><60\mu\text{A}</math>)
- Idle (lowest run mode)
- Run mode (dominated by the frequency of MCU operation and minimizing the use of active functional device peripherals)

The MC1322x power management is controlled through the Clock and Reset Module (CRM). The CRM is a dedicated module to handle MCU clock, reset, and power management functions that includes control of the power regulators. All these functions have impact on attaining lowest power.

Figure 1 shows the 1322x-LPN block diagram for the 1322x-LPN reference design.

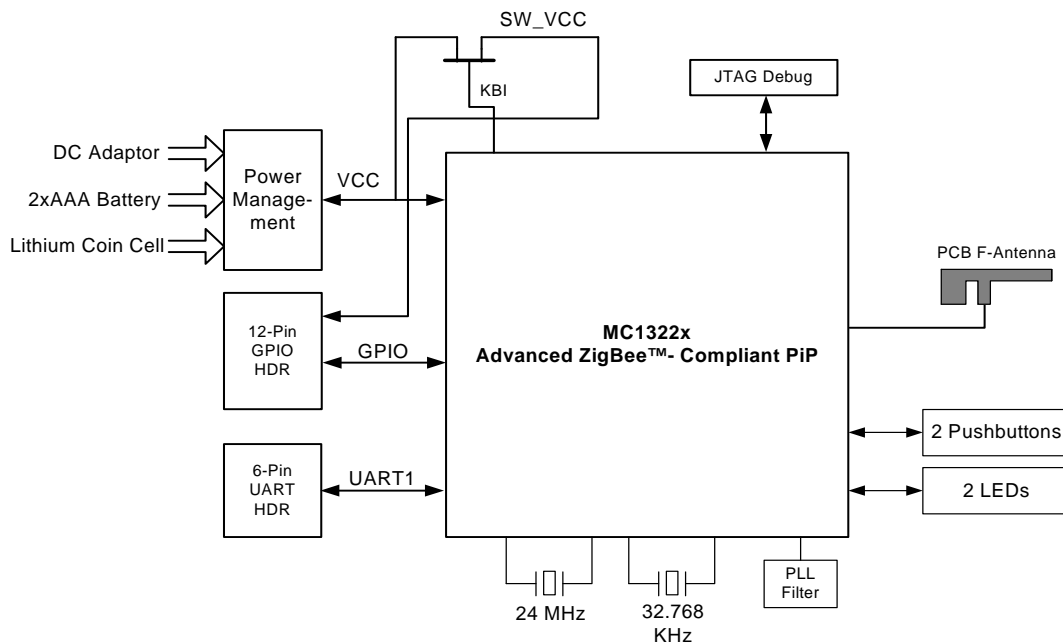


Figure 1. 1322x-LPN Block Diagram

## 2.1 CRM Features

In addition to other features and for our purposes, the CRM low power features include the following:

- Control system reset
- Control clock gating for power savings
- Sleep mode (Hibernate and Doze) management
  - Degree of chip power down

- Retention of programmed parameters
- Programmable retention of RAM contents
- Clock management
- Wake-up management
  - Graceful power-up
  - Clock management
  - Wake-up via programmable timer or external interrupts.
- Wake-up timer
  - Hibernate mode - based on onboard 2 kHz oscillator or optional 32.768 kHz crystal oscillator
  - Doze mode - based on main reference oscillator, typically 24 MHz
- Controls reference clocks based on default 24 MHz crystal oscillator or optional 13-26 MHz oscillator with PLL for 24 MHz frequency synthesis.
- MCU watchdog timer (COP)
- Software initiated reset
- Management control of onboard linear regulators and optional buck regulator

## 2.2 CRM Operation

The CRM has primary control of the entire system:

- Reset and power up — After release of the hardware RESETB signal, the CRM will perform a power up sequence of the MCU after the POR. The linear regulators and clock sources are managed for a graceful start-up of the MCU and its resources. The radio is not powered until needed
- Normal operation of MCU — The clock management of the MCU and its resources are controlled by the CRM. The processor clock is programmable from low frequencies up to the maximum reference frequency (13-26 MHz optional w/24 MHz standard) to allow the application to trade-off processing speed versus power savings
- Sleep modes and recovery — There are two sleep modes of Hibernate and Doze. The primary difference is that Doze mode keeps the reference oscillator running. Both modes can retain critical programmed parameters and have selectable sizes of RAM retention. Hibernate has lowest power, but Doze allows high accuracy sleep timing. The CRM manages the recovery from low power, similar to power-up from reset, providing regulator and clock management.
  - Wake-up can be based on external interrupts through 4 KBI inputs
  - Wake-up can be from internal interrupts
  - Wake-up can be based on an RTI (wake-up) timer.
- The RTI timer with 2 possible frequency sources provides a very low power wake-up option from sleep
  - One option is a onboard, low accuracy 2 kHz oscillator
  - A second option is to add an external 32.768 kHz crystal for the RTI clock source
  - A 32-bit timer allows greater than a 36.4 hour wake-up delay with the 32.768 crystal oscillator

- Other features of the CRM:
  - An optional COP watchdog timer to monitor CPU program activity
  - A programmable software reset

## 2.3 Low Power Considerations when using the 1322x-LPN

In addition to setting the application into hibernate or doze mode, the developer must consider the following when designing for low power operation:

- Indicator LEDs connected to ground not VCC, eliminating GPIO state high during low power mode.
- The GPIO connector (J2) as defined on the 1322x-LPN provides access and connectivity to the MC1322x
  - SW\_VCC is provided as a switched supply via KBI\_3. Current draw should be limited to 50 mA.
  - No peripheral device should pull a GPIO high during low power mode
  - Signal SWITCH1 is connected to KBI\_4 and can be used as a wake-up interrupt to the MC1322x (See the CRM chapter of the *MC1322x Reference Manual*).
  - Signal SWITCH2 is connected to KBI\_0 and can be used as a wake-up signal from the MC1322x to an external device. (See the CRM chapter of the *MC1322x Reference Manual*).
- The UART1 connector (J3) provides direct access to the MC1322x GPIO. If low power mode is enabled and UART1 is connected to an external device, extraneous high leakage current can occur if any of the UART1 signals is driven high.

As mentioned, for low power operation the static RAM retention is selectable for both hibernate and doze modes and are summarized

- 96 Kbytes RAM.
  - RAM0: 8 Kbytes, 2 Kwords (2048 x 32 bits)
  - RAM1: 24 Kbytes, 6 Kwords (6144 x 32 bits)
  - RAM2: 32 Kbytes, 8 Kwords (8192 x 32 bits)
  - RAM3: 32 Kbytes, 8 Kwords (8192 x 32 bits)
- All read or write accesses require a minimum of two system clock cycles
- Stall signal generated for read after write cycles
- Clock is enabled on the accessed memory device only for low power consumption
- RAMs have been divided to allow for power savings. While sleeping, combinations of the 4 RAMs can be turned off and the RAM remainder can be placed in a low voltage mode for data retention. If more RAMs are turned on, then less battery life will be achieved. Depending on the amount of RAM powered during sleep, the boot time may be longer with less RAM as the non-powered RAM must be reloaded from FLASH.
- For battery based applications, an optional buck regulator is provided to maximize battery life. The buck regulator drops the higher battery voltage to 1.8 - 2.0 Vdc that is applied to the onboard linear regulators allowing lower net current from the battery maximizing its life.

## NOTE

- This discussion is highly concerned about minimum current usage. Although power is actually the product of voltage times current, the MC1322x has extensive onboard voltage regulation that tends to minimize current variation due to power supply (VBATT or VDD) variation. Also, battery capacity is typically rated as available MAH (milliampere-hours) due to the variance of battery voltage due to lifetime, use model, and temperature.
- There are many options available for low power operation. These are discussed in this application note, however, it is the user's responsibility to properly apply any of these to their unique application. The user is directed to the MC1322x Reference Manual for details of implementing the options discussed here.

## 2.4 Sleep Mode (Low Power) Overview

Depending on use model, the low power mode can, perhaps surprisingly, dominate overall current usage. The MC1322x provides a number of low power mode options and attention to GPIO usage affect low power current:

- All GPIO must be configured for low power operation
- Some GPIO may need to be re-configured when entering/exiting low power mode
- Wake-up is possible from -
  - Reset from external request
  - Pushbutton operation via KBI (Keyboard Interrupt Module) interrupt request
  - Timer operation using a wakeup timer or Real Time Counter (RTC)

## 3 Configuring MC1322x GPIO for Best Practice

The 1322x-LPN reference design is intended to develop low power operation and applications code. It has been designed to allow low current measurement and support MC1322x low power application needs. The MC1322x by default does not retain power to its GPIO pads while in low power mode (KBI signals are the exception and do retain power). As a result, GPIO pads should not be driven high while the device is in low power or extraneous current can exist. The 1322x-LPN provides a switched VCC for off-board devices connected to the MC1322x while in low power mode so that these devices can be disabled when appropriate.

Best practice design demands that the user configure all device GPIO to a known condition.

- Unused GPIO should be configured either as an input with a pullup/pulldown or an output in the low state
- Putting GPIO in a known condition prevents a GPIO from “floating” as an input with an unknown state, which can result in undesired excess “shoot-through” leakage current.
- GPIO used as peripheral functions may need re-configured when entering low power and then re-initiated for peripheral use after wake-up/recovery for run time.

**NOTE**

- Be aware that the boot process can affect the state of the GPIO and leave a GPIO in an altered state. The impact of the boot process on the GPIO is discussed in Section 3.12.7 of the *MC1322x Reference Manual*.
- When first writing software for low power operation, users can sometimes find that the actual device current usage is in the 10's of  $\mu\text{A}$  versus expected 1-2  $\mu\text{A}$  or less. When this occurs, the most probable cause is some unexpected condition on an input that is causing “shoot-through” current or driving the input to a state where current is being drawn through a pullup/pulldown.
- It is strongly suggested that the user review the low power considerations described in the *MC1322x Reference Manual*.

### 3.1 GPIO Pin State for Reset, Default, and Low Power Modes

Table 1 shows the state of the GPIO signals versus mode of the device. **The user is strongly advised to review this information.**

Reset and low power operation must consider the following points:

- With RESETB active or in Hibernate or Doze, all GPIO except the KBI signals are unpowered (disconnected from VBATT).
- It is best practice not to drive a GPIO high or tie a GPIO high through a resistor when the GPIO is unpowered.

**NOTE**

Driving a GPIO high when not powered can forward bias the ESD diode and cause excess current. No damage will be done to the device, but excess leakage current can be high and defeat the desire for very low current while in a low power mode.

- The KBI signals are powered during reset, Hibernate, and Doze
  - when reset is active all the KBI revert to inputs pulled low
  - When in low power Hibernate or Doze mode, 4 KBI are outputs to disable external circuitry and 4 KBI are inputs to allow an external asynchronous interrupt to wakeup the device

### 3.2 Using GPIO in Low Power Modes

The MC1322x was designed to have extremely low current during low power modes. Because of the large number of GPIO on the device, the GPIO are generally disconnected from VBATT when a low power mode is enabled, with the exception of the KBI signals as described in Section 3.1, "GPIO Pin State for Reset, Default, and Low Power Modes". It is envisioned that the MC1322x will be generally used in two basic configurations:

- As a peripheral communication channel for a primary device such as a cell phone.
- Standalone (perhaps with its own peripheral functions), where the MC1322x is the primary device

### 3.2.1 MC1322x as a Peripheral Device

In a battery operated device such as a cell phone, the peripheral function must use as little power as possible, especially when disabled and not in use. For this scenario, the reset condition is a desired low power state because of absolute lowest current. Figure 2 shows a simple interconnect diagram for this application.

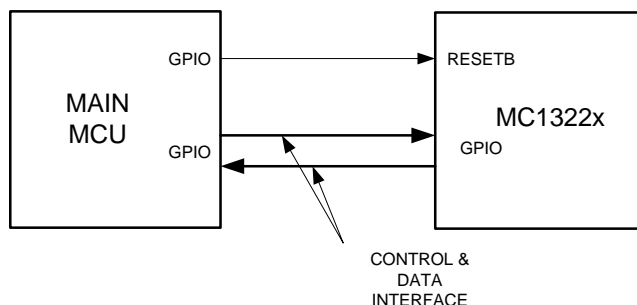


Figure 2. MC1322x as a Peripheral Function

The MC1322x RESETB should be driven from a main MCU GPIO that is always active. When RESETB is active all MC1322x GPIO (except KBI) will be unpowered and need not be tied to any voltage.

- It is best that other peripheral devices not be tied to the MC1322x.
- The control and data interface between the devices would typically be a SPI, SSI, UART, or other serial interface.
  - MC1322x normal outputs (when reset active) revert to no power and will not drive the main MCU inputs. The main MCU may either provide a pull-down under this condition or be over-programmed as outputs in the low condition to hold the MC1322x signals low.
  - The MC1322x inputs should be driven low by the main MCU outputs.
  - If the I<sup>2</sup>C is used as the data interface, any required external pull-up resistors must be disabled when the RESETB is active.
- The disadvantage to using reset as a low power condition is the recovery time of the MC1322x. The device must recover from reset, start its clocks, and load its RAM to be able to be active.
- Before driving the MC1322x to a reset condition after the device has been active, the host should insure that important data or parameters on board the MC1322x be saved in non-volatile memory for later recovery.

### 3.2.2 MC1322x as Primary Device

The more common scenario is where the MC1322x is the primary device and may have peripheral devices such as sensors tied to it. In this situation reset is not typically used as a low power condition. Figure 3 shows examples of how external connections can be controlled.



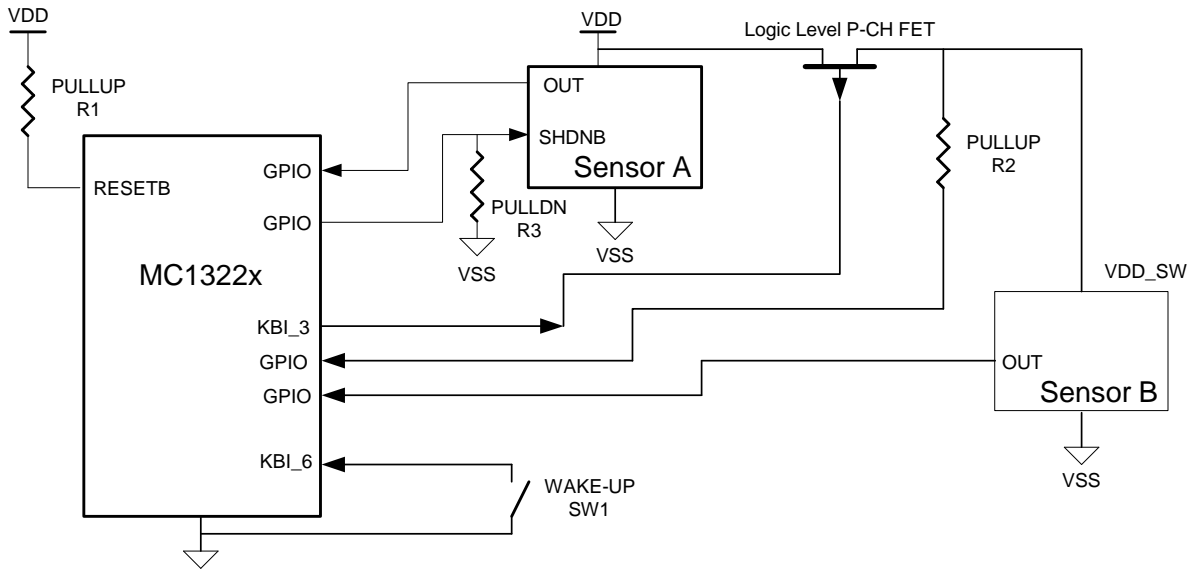


Figure 3. MC1322x as a Primary Device

- A pull-up resistor should be connected to RESETB.
- If the peripheral device has a shutdown input (active low) such as Sensor A, then a standard GPIO can be used to disable the device. The external pull-down is required to be sure the SHDNB input is pulled low in the reset or low power state.
- Other external connections such as pull-up resistors or a device without a shutdown option (Sensor B) can be controlled by an external P-channel MOSFET. A logic level gate voltage device is recommended, and the gate can be driven from KBI\_3 through KBI\_0. These outputs retain their state under low power states and default to a high condition to keep the FET off during reset (and Hibernate and Doze if desired).
- For a Hibernate or Doze wakeup option, an external switch can be connected to KBI\_4 through KBI\_7.

### 3.2.3 Using KBI Signals as Pushbutton Interface

The KBI signals are unique because they are the only GPIO that normally remain powered during low power operation defined by the following:

- KBI\_7:KBI\_4 are always inputs
  - They are the only GPIO inputs that can generate an interrupt request from an external transition on the input (either low power or normal operation).
- KBI\_3:KBI\_0 are always outputs

A common use is as a keypad or pushbutton array interface. With only four inputs and four outputs, a square matrix of 16 keys can be supported. This is the max number of switches that can be supported for low power mode with wakeup capability.

## Configuring MC1322x GPIO for Best Practice

If a larger number of keys is required a non-square array of N columns vs. 4 rows may be used. Additional non-KBI GPIO can be used as column drivers against the normal 4 row KBI signals. The added columns cannot provide wakeup during low power as they are not powered during powered.

### NOTE

- For additional keypad configurations, see Section 3.4 of the *MC1322x Reference Manual*.
- For proper use of the event-driver interrupt requests on KBI\_7:KBI\_4, see Section 5.9.2 of the *MC1322x Reference Manual*. The interrupts are always controlled via the CRM. However, during normal operation all the KBI pads are controlled by the GPIO Controller and must be programmed as accordingly.

### 3.2.4 External Clock Connections

The MC1322x uses clock sources for two primary functions:

- Reference clock oscillator - all active clocks for the radio, CPU, and peripherals are derived from the reference oscillator.
- Real Time Clock (RTC)- wakeup timer for Hibernate or Doze modes. The RTC input clock can be derived from three sources:
  - 2 kHz internal ring oscillator
  - Reference oscillator divided by 128 (nominally 24MHz/128)
  - Optional external 32.768 kHz crystal oscillator

### NOTE

The MC1322x external clock connections are described in more detail in Section 3.5 of the *MC1322x Reference Manual*.

### 3.3 GPIO Pin Default State Out of Reset

[Table 1](#) lists the conditions of the GPIO as the MC1322x device exits a hardware reset. The boot process follows a hardware reset.

### NOTE

Be aware the boot process can effect the state of the GPIO leaving it in an altered state. Program flow including recovery from other low power modes is shown in Chapter 3.12.3 of the *MC1322x Reference Manual*.

Table 1. GPIO Pin Default State

Item	Pin Function	Pin Name	Default Function	Reset Active	Default State (Upon exiting reset, before Boot)	Hibernate & Doze
1	Standard GPIO <sup>1</sup>		GPIO	Unpowered	Inputs w / pull-downs enabled and bus keepers disabled	Unpowered
2	I <sup>2</sup> C Bus	I2C_SDA, I2C_SCL	GPIO	Unpowered	Inputs w / pull-ups enabled and bus keepers disabled	Unpowered
3	ADC Reference Voltage	ADC1_VREFH, ADC1_VREFL	Analog	Unpowered	Analog Reference Voltage Inputs (no pull-up/pull-down and bus keeper disabled)	Unpowered
4	ADC Reference Voltage	ADC2_VREFH, ADC2_VREFL	GPIO	Unpowered	1) ADC2_VREFH - Input w / pull-up enabled 2) ADC2_VREFL - Input w / pull-down enabled <sup>2</sup> 3) Pad keepers enabled for both pins	Unpowered
5	ADC Analog Input Channel	ADC6 - ADC0	GPIO	Unpowered	Inputs w / pull-downs enabled and bus keepers enabled	Unpowered
6	ADC7_RTCK Debug Port Clock	ADC7_RTCK	RTCK	Unpowered	JTAG RTCK output	Unpowered
7	JTAG	TDI, TMS, TCK, TDO	JTAG	Unpowered	1) TDI, TMS, and TCK - Input w/pull-ups enabled and pad keepers disabled 2) TDO - Output	Unpowered
8	Nexus (includes JTAG and RTCK)	MDO07-MDO00, MSEO0_B, MSEO1_B, MCKO, EVTO_B, EVTI_B, RDY_B	Nexus	Unpowered	All Nexus signals are outputs with the exception of the input EVTI_B w / pull-up enabled	Unpowered
9	KBI	KBI_7 - KBI_0	GPIO	Powered (revert to default CRM KBI control)	1) KBI_3:KBI_0 - outputs high 2) KBI_7:KBI_4 - inputs w/pull-downs enabled	Powered (revert to CRM KBI control)

<sup>1</sup> "Standard GPIO" refers to all GPIO not listed as special cases in this table.

<sup>2</sup> ADC2\_VREFL exits reset with pull-up enabled, but boot code changes to pull-down enabled for "clear FLASH" test. See Section 3.12.5 of the of the *MC1322x Reference Manual*.

## 4 Minimizing Run Time Current

As would be expected, greater current draw is the result of the device run time mode, and as with any CMOS device, the current draw is highly dependent on the frequency of operation. There are two primary strategies then for minimizing run time current:

- Disable as much logic use as possible - The MCU peripheral current draw is most affected by enabling/disabling required peripherals, especially the ADC module
- Control CPU and peripheral clock rates where possible - The CPU base current is typically about 3.5 mA with a CPU/peripheral clock of 24 MHz

In the following paragraphs discuss briefly these strategies (Refer the *MC1322x Reference Manual*).

### 4.1 Reducing CPU Clock Frequency

When the MC1322x is not in sleep mode there are strategies that can be used to minimize run-time power. One means to reduce run time current draw is to reduce the CPU clock if the system performance or peripheral timing does not require the higher clock rate. The transceiver is capable of full operation with reduced CPU clock. The clock can sometimes be slowed down if the application does not require high throughput.

#### NOTE

- Reduced clock rate for radio operation is not recommended operation because the software may need to respond to an abnormal event; keeping the CPU clock at 24MHz for radio operation is suggested and use of bus-steal is recommended as a better choice.
- If the radio is active, i.e., an RX or TX sequence is to be in progress, there is a minimum 2 MHz required clock rate for the MACA and DMA to work properly.

### 4.2 Disabling MCU Peripherals During Run-time

The MC1322x provides an extensive set of peripherals and current can be saved by disabling any peripheral when unused or not needed. Run-time current is the actual operating current of the MC1322x when in normal operation, not sleep mode. By understanding how operation is controlled, current usage can be predicted and minimized. Refer to Chapter 3.10.4 of the *MC1322x Reference Manual* for more information.

#### NOTE

All peripherals can be enabled individually. Enable peripherals only as required. One caution is that a given peripheral may require a higher peripheral clock rate than would be set a minimum required rate during TX or RX.

## 4.3 Managing Transceiver (Radio) Operation

The MC1322x transceiver consumes the greatest share of current when transmitting or receiving. Control of the radio is usually done through the codebase/stack services, but careful manipulation of the transceiver/modem control may save some current:

- Transmit time is very deterministic due to known packet length, and as a result, little or no control of the radio is possible to reduce TX “on” time
- Receive “on” time is typically more dominant due to not knowing the exact time an incoming packet will arrive. For a particular use model, review when the RX mode is enabled. Consider if the RX mode “on” time can be reduced; do not just leave RX continually on as standard practice.
- The use and configuration of the RF modes and control signals can initially be set in the BeeKit™ Wireless Connectivity Toolkit via the “Platform Editor” of its “Project Configuration” tool (see Chapter 3.9 of the *MC1322x Reference Manual*). Once configured these options will be retained when initialized from reset and recovering from a low power mode.
- Enable bus-steal module for active radio sequences - The MC1322x has a unique bus-steal module to minimize MCU current drain during radio operation and save overall power. The MCU data bus has three bus masters that include the MACA DMA function (for moving packet data during a RX or TX), the CPU core, and the bus-steal module. The DMA works on a cycle-steal basis and has highest priority (it is also the reason there is a minimum required clock rate during an active radio sequence).
- Disable the analog regulator for lowest power (same operation for low power mode) - the analog regulator, as standard practice, is enabled during stack/transceiver initialization. If the radio is not in use for an extended period of time, then the regulator can be disabled, and then re-enabled prior to radio use.

### NOTE

Caution must be observed if the analog regulator is disabled while the stack is active. The user must prevent the stack from trying to use the radio while the analog regulator is disabled.

## 4.4 Means of Wake-up from Low Power

Upon exiting reset, Hibernate or Doze modes, the MC1322x will jump to Idle mode where the MCU will be fully functional and operates from the reference oscillator. Communication with all mapped registers is possible including the digital modem and RF analog control. The transceiver is completely off. The Sleep module of the CRM gracefully controls this transition.

The following describes the basic mechanisms for wake-up from sleep modes:

- Reset - Upon release of the hardware reset signal, the device will wakeup and boot from a cold start condition.
- Timer-based - There are two separate timers available that can generate a wakeup interrupt
  - Wakeup timer - This timer is a 32-bit counter that can be clocked from the 2 kHz ring oscillator, the reference oscillator divided by 128, or the 32.768 kHz oscillator. The source is selected as a sleep option.

## Summary Flow for Entering Low Power Mode

- Real Time Clock (RTC) - This is a second 32-bit timer that can be clocked from the 2kHz ring oscillator or the 32.768kHz oscillator.
- KBI input transition - Four KBI signals (inputs KBI\_7:KBI\_4) remain powered during sleep mode and a transition on the input can be enabled to generate an asynchronous external event (such as a button push) to wakeup the device.

## 5 Summary Flow for Entering Low Power Mode

The following flow summarizes actions for entering low power mode:

1. Exit any stack/application gracefully - save any required parameters, shutdown any tasks that are affected.
2. As required reconfigure any GPIO - it is assumed that during system initialization, any unused GPIO are configured properly so as to not cause excess leakage current. At this point, re-configure any GPIO in use that may special handling during low power so as to not cause excess leakage - see [Section 3.1, "GPIO Pin State for Reset, Default, and Low Power Modes"](#)).
3. Disable the transceiver analog regulator if required - see [Section 4.3, "Managing Transceiver \(Radio\) Operation"](#).
4. Disable any active peripheral clocks.
5. Initiate any required wake-up sources / interrupts - choosing means and clocks as previously recommended
6. Set sleep mode conditions in the Sleep Control (SLEEP\_CNTL) Register - See the *MC1322x Reference Manual, Chapter 5.9.3*.
7. Set the DOZE or HIB in SLEEP\_CNTL register and wait to the SLEEP\_SYNC bit to be set.
8. Once set, clear the SLEEP\_SYNC bit. This will complete the power-down process of the MCU.

### NOTE

The CRM\_GoToSleep() API places the IC in sleep mode (either doze or hibernate) and configures the sleep mode parameters. Users may use other CRM driver APIs to properly set the wakeup sources and other sleep mode configurations. Please refer to the *MC1322x Software driver reference manual (22xDRVRRM.pdf)* for further details about the APIs available.

## 6 Managing Low Power from Freescale Software Stacks

For purposes of this application note, the SMAC software stack has been use to demonstrate low power design for the MC1322x IC since BeeKit provides an out-of-the-box demo as a software starting point, and also Freescale provides the 1322x-LPN reference design hardware for a simple turn-key customer approach.

Low power applications however can be designed for use with software stacks built on top of the Freescale 802.15.4 MAC, such as BeeStack Consumer (RF4CE), SynkroRF, BeeStack 2007 and the 802.15.4 MAC itself managing low power by using the *Low Power Library* platform module. The *Low Power Library*

provides a common way of configuring low power across software stacks and hardware platforms, as well as a common API set for managing low power entry and exit from an application viewpoint.

Managing low power from a software application perspective happens in two steps:

- The first step involves how an application is readied to enter low power, and subsequently exit low power (ready to re-enable the stack). This step differs across software stacks.
- The second step is to then call Low Power Module APIs to place the hardware platform into a low power mode. Again, this step differs across software stacks.

## 7 Sample Application Demonstrating Low Power

This note is supplemented by BeeKit application that demonstrates an example of low power operation; the Low Power Bell demo. This application is intended to demonstrate the steps of Chapter 5 with minimal code modifications and put the MC1322x into low power modes illustrating the respective current. The Low Power Bell application target platform is the 1322x-LPN and it must be configured properly to observe the operating current. The schematic for the 1322x-LPN can be found in Chapter 4 of the 1322x Low Power Node Reference Manual (1322xLPNRM.pdf). Using BeeKit, the application source code can be created so that the user can simply configure the hardware, load the executable, and observe the current. See [Section Appendix A, “Sample Application Demonstrating Low Power”](#) for a listing of the C source code.

### 7.1 Configuring the 1322x-LPN Hardware

The power management and measurement circuitry for the module must be configured properly to supply power to the module as well as measure the device operating current. [Figure 4](#) and [Figure](#) show the power management circuitry and points of reference for the module:

- DC voltage to the module is supplied by the following sources:
  - Regulated 3.3V DC source (4.4 to 12 VDC input supply). When connected, it automatically shuts down the battery supply.
- AAA battery source with a typical usable voltage range for 2 AAA cells is ~3.1V down to 2.2 - 2.0V. Note that a Schottky diode and a switch MOSFET are in series with the battery pack, and these will impact the lowest usable voltage for the battery pack.
  - An optional buck switched regulator can be used for longer battery life.
- Lithium coin cell source. For very low duty cycle applications where the system mostly “sleeps” in low power.
  - Jumper J6, Pins 2-3 must be connected

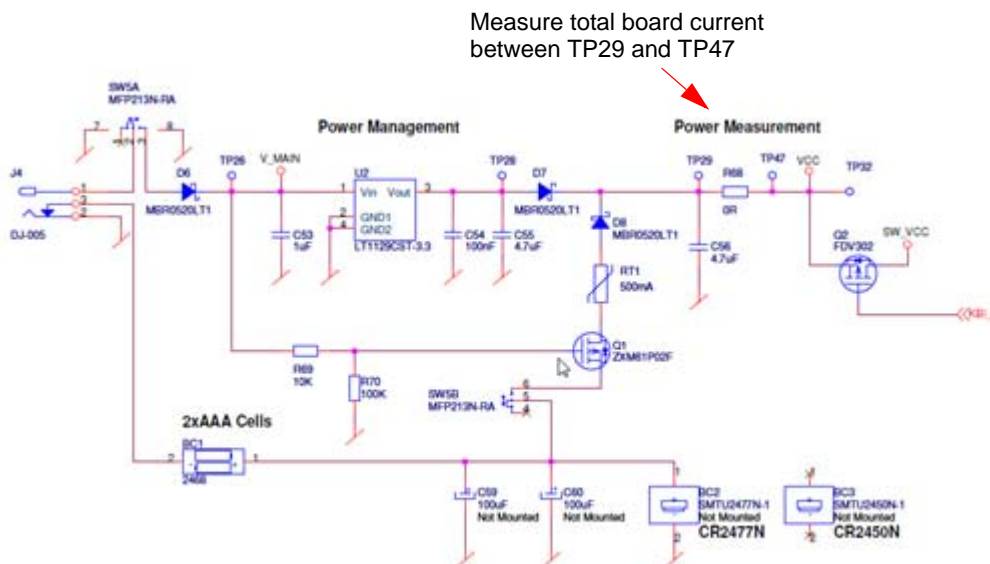


Figure 4. 1322x-LPN Power Management Circuit

The 1322x-LPN connectors, push buttons and LEDs are shown in [Figure 5](#) allowing for easy access and setup for low power current measurements.



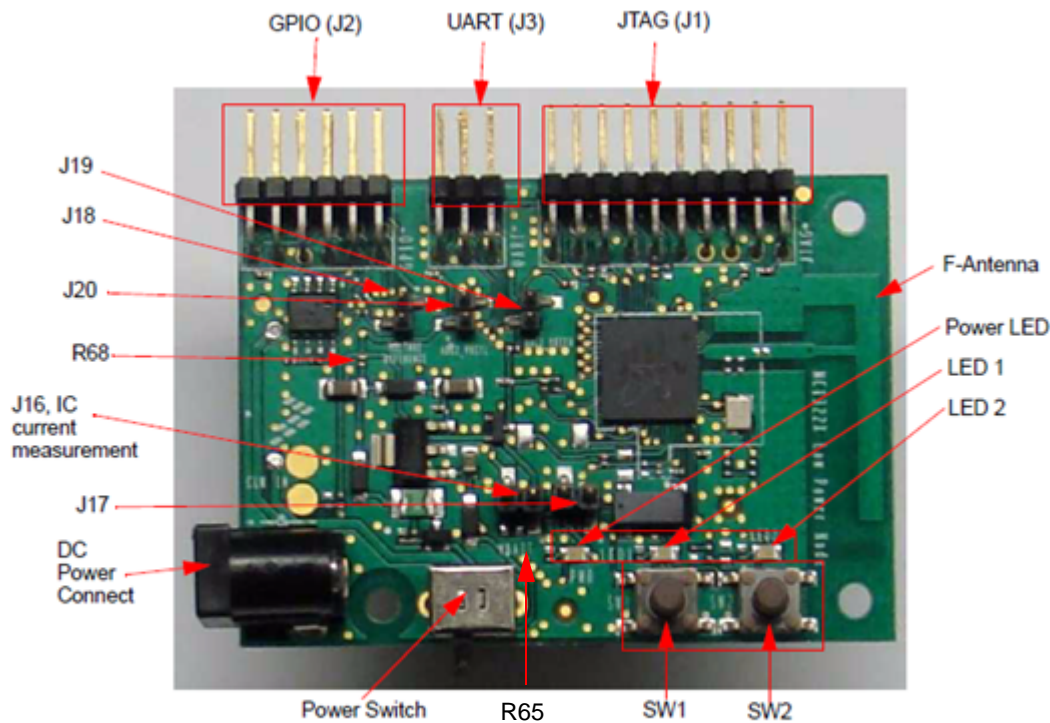


Figure 5. 1322x-LPN Connectors, Push Buttons and LEDs

Figure 5. 1322x-LPN Connectors, Push Buttons and LEDs

Three primary points for current measurement are:

- Total current to the MC1322x IC. R65 can be removed and current can be measured through jumper J16. [Figure 6](#) shows example circuit.
- Total current to the board. R68 can be removed and current can be measured between TP29 and TP47. [Figure 7](#) shows example circuit.
- ADC voltage reference current. Voltage reference U17 is enabled via J18; current can be measured through this jumper. [Figure 8](#) shows example circuit.

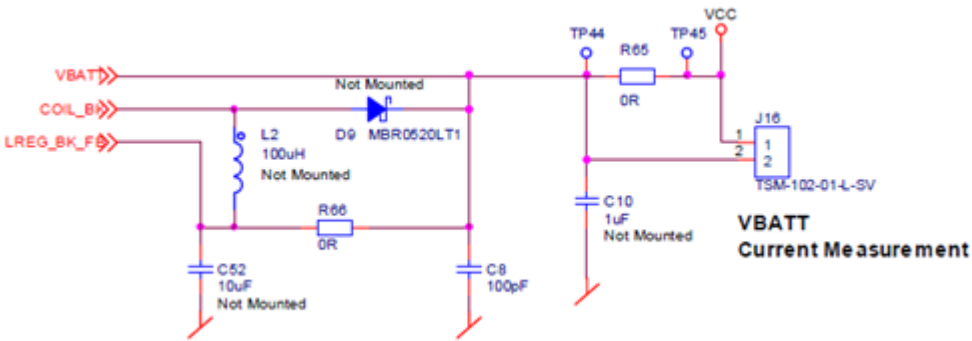


Figure 6. Total current to MC1322x IC measurement through J16

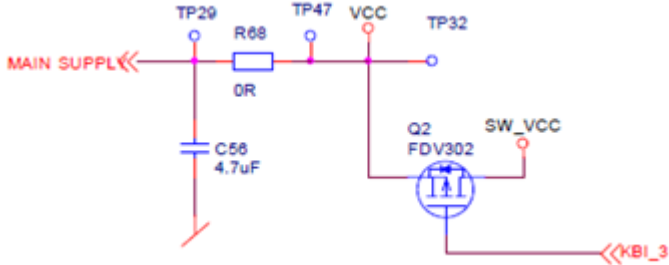


Figure 7. Total board current measurement between TP29 and TP47

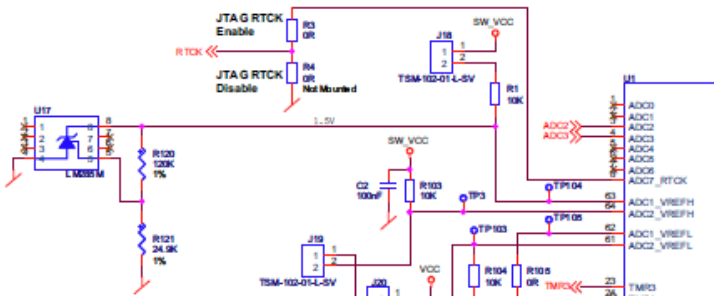


Figure 8. ADC voltage reference U17 enabled via J18

## 7.2 Running the SMAC Sample Application to Measure Low Power on the 1322x-LPN Module

Create a project using BeeKit as described in Appendix A for the Low Power Bell application. It is assumed the user is familiar with the Freescale development tools for the MC1322x. To simply run the low power app:

1. Attach the debug tool to the boards JTAG port at J1 connector
2. Apply a DC supply to the 1322x-LPN
3. Set current meter as required at J16 (R65 removed)
4. Download “Low Power Bell TX.bin” using IAR Embedded Workbench IDE or TestTool
5. Remove JTAG tool from J1
6. Press the reset button on the 1322x-LPN
7. Observe current meter (should be approximately 0.9 - 5.1uA hibernate current). The Low Power Bell demo consists of two applications: the transmitter, which is continuously in low power mode until a button is pressed, then it sends a message indicating that the button was pressed. The receptor is monitoring and waiting for a packet, when it receives a valid packet plays a melody.

## 8 Summary

This application note gives a detailed description of areas that must be addressed to provide lowest power when using the MC1322x transceiver. No attempt is made to provide examples for all possible low power combinations, but these guidelines can be applied to all combinations. Through the BeeKit software configuration based GUI and APIs Freescale provides, building simple demo applications is made easy for the user.

To assist in programming low power applications code, Freescale provides the 1322x Low Power Node with SMAC-based software for the MC1322x transceiver as well as Freescale’s IEEE 802.15.4 based stacks for other platforms. A sample stack-based application (Low Power Bell) with source is provided to demonstrate low power modes and give a starting example for user applications. The Low Power Bell TX application waits in low power mode until a button is pressed, then it wakes up and sends a frame to the Low Power Bell RX application. More information on the Low Power Bell demo can be found in chapter 5 of the *MC1322x SMAC Demonstration Applications User’s Guide (22xSMACDUG)*.

In addition to the low power principles described and application demo, an optional buck regulator is provided to maximize battery based application life. This allows lower net current from the battery to maximize the life of the battery.

# Appendix A Sample Application Demonstrating Low Power

To demonstrate low power capabilities of the MC1322x devices, Freescale provides both software and hardware for this specific purpose. Freescale offers application templates supporting low power features such as the SMAC Low power bell TX demonstration application. The application target platform used for this demo is the MC1322x-LPN and has been designed to allow low current measurement and support MC1322x low power application needs by using the minimum peripherals required in order to achieve low power implementation.

The MC1322x-LPN hardware must be configured properly to observe the lowest operating current. Users must also be aware that any external peripherals and respective software configuration may cause current leakage so care must be taken to avoid this.

## A.1 Generating an SMAC Low Power sample Application

The Low Power Bell Demonstration consists of a transmitter (TX) and receiver (RX). The Low Power Bell TX application serves as the bell switch and demonstrates low power modes and how to implement them. This section will discuss how to generate and run the transmitter (TX). Users may refer to the *MC1322x SMAC Demonstration applications User's Guide (22xSMACDAUG.pdf)* for further details about generating the RX.

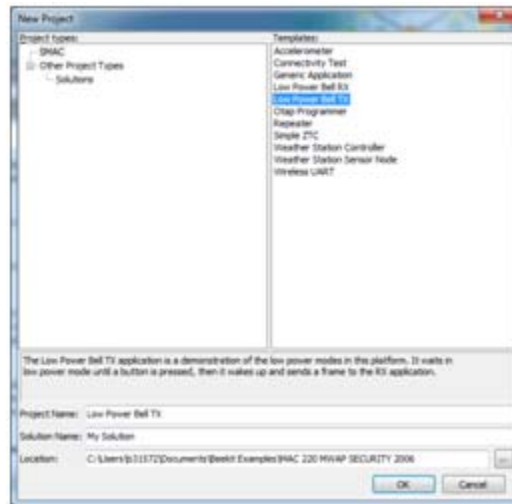
To generate the Low power bell TX project from BeeKit:

1. Open BeeKit (Start->Programs->Freescale BeeKit->Freescale BeeKit).
2. In BeeKit, select the latest ARM7 SMAC codebase (File menu -> Select Codebase...)



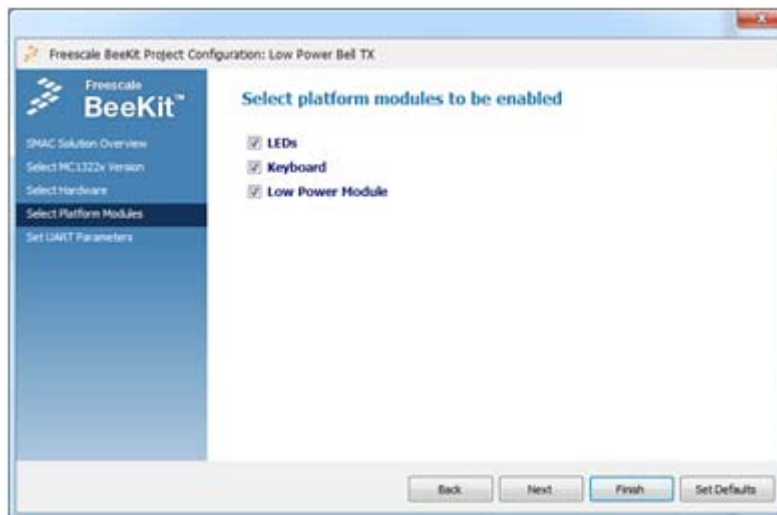
Figure 9. Selecting ARM7 SMAC Codebase from BeeKit

3. Click on File -> New Project and choose the Low Power Bell TX template.



**Figure 10. Create a new project using Low power bell TX template**

4. Follow the steps as displayed in the New Project wizard. Make sure to select the MC1322x Low power node (1322xLPN) as the target board and to enable the low power module.



**Figure 11. Configure the application properties as needed**

5. Finally, validate and export the solution to IAR IDE. Make sure you are using the proper IAR version needed. Refer to the ARM7 SMAC codebase release notes included in Freescale installation folder (i.e. C:\Program Files\Freescale) under System Requirements.

**NOTE**

For further details about BeeKit and how to generate solutions based in SMAC or any of Freescale codebases, please refer to the BeeKit User's guide.

## A.2 Low power configuration options

Device usage is loosely divided into low power (sleep) options and normal run options. The MC1322x allows for a wide variation of low power options, however, the most basic modes include:

- Reset - The MC1322x can be held in a reset state via the hardware reset pin. This has the absolute lowest power, but the longest recovery time.
- Hibernate - Has the lowest power (excluding reset), but does not have the reference oscillator running. Exit from Hibernate is via timers driven by low power oscillators (a default 2 KHz low accuracy oscillator or an optional 32.768 kHz crystal oscillator) or KBI asynchronous interrupt requests.
- Doze - Very similar to Hibernate, however, Doze keeps the reference oscillator running. It has the advantage of a low cost accurate time base (no 32 kHz crystal required) while in sleep condition although at the cost of additional current.

The MC1322x can enter into Hibernate or Doze modes depending the users application requirements. By default, the Low Power Bell TX application uses Hibernate mode but users may change to doze by simply changing the MLME function call to `MLMEDozeRequest()` from `MLMEHibernateRequest()`.

Typical example functions calls are as follows:

- `MLMEHibernateRequest(gRingOsc2khz_c, SleepCtl);`
- `MLMEDozeRequest(SleepCtl)`

In addition, the user must consider the impact of recovery options when from low power modes. For example, a low power option configured for wake up timing can also impact current consumption during the low power state. This must be considered by the user. There is a trade-off between lowest power and faster wakeup time (from sleep to active radio).

Especially important is:

- Amount of RAM retained - when returning from low power mode, the recovery time will be greatly impacted by how much RAM content needs reloaded.
- MCU retention - when returning from low power with MCU retention enabled, the CPU will start from the condition or state at which it stopped and will continue. If retention is disabled (advantageous for lower power), the CPU will start executing from the bottom of RAM as in a start-up condition. Software must be able to recover properly, and good practice may be to save an image of system status (in retained RAM) before entering sleep.

## A.3 Amount of RAM Retained

The amount of RAM retained during low power mode is configured in the `PWR_Config.h` file of the Low power Bell TX project as follows:

```
// To configure the RAM retention mode used during sleep
// gRamRet8k_c: First 8k of RAM retained during sleep
// gRamRet32k_c: First 32k of RAM retained during sleep
// gRamRet64k_c: First 64k of RAM retained during sleep
// gRamRet96k_c: First 96k (All) of RAM retained during sleep
```

```
#define gRAMRetentionMode_c gRamRet96k_c
```

## NOTE

Current consumption values when retaining an additional blocks of RAM are stated in MC1322x datasheet (MC1322x.pdf)

### A.4 MCU Retention

MCU retention could be either enabled or disabled in PWR\_Config.h file by modifying:

```
// To configure the MCU retention mode used during sleep
// 0: No MCU retention
// 1: MCU retention is On during sleep
#define gMCURetentionMode_c TRUE
```

### A.5 MCU Retention

As with the MCU retention, the same principle applies to the MCU Pad retention. Pad retention will only be retained if MCU retention is set true.

```
// To configure the MCU Pad retention mode used during sleep (if MCU retention configured)
// 0: No Pad retention
// 1: MCU Pad retention on id MCU retention was configured also
#define gMCUPadRetentionMode_c TRUE
```

**How to Reach Us:****Home Page:**

[www.freescale.com](http://www.freescale.com)

**Web Support:**

<http://www.freescale.com/support>

**USA/Europe or Locations Not Listed:**

Freescale Semiconductor, Inc.  
Technical Information Center, EL516  
2100 East Elliot Road  
Tempe, Arizona 85284  
+1-800-521-6274 or +1-480-768-2130  
[www.freescale.com/support](http://www.freescale.com/support)

**Europe, Middle East, and Africa:**

Freescale Halbleiter Deutschland GmbH  
Technical Information Center  
Schatzbogen 7  
81829 Muenchen, Germany  
+44 1296 380 456 (English)  
+46 8 52200080 (English)  
+49 89 92103 559 (German)  
+33 1 69 35 48 48 (French)  
[www.freescale.com/support](http://www.freescale.com/support)

**Japan:**

Freescale Semiconductor Japan Ltd.  
Headquarters  
ARCO Tower 15F  
1-8-1, Shimo-Meguro, Meguro-ku,  
Tokyo 153-0064  
Japan  
0120 191014 or +81 3 5437 9125  
[support.japan@freescale.com](mailto:support.japan@freescale.com)

**Asia/Pacific:**

Freescale Semiconductor China Ltd.  
Exchange Building 23F  
No. 118 Jianguo Road  
Chaoyang District  
Beijing 100022  
China  
+86 10 5879 8000  
[support.asia@freescale.com](mailto:support.asia@freescale.com)

**For Literature Requests Only:**

Freescale Semiconductor Literature Distribution Center  
1-800-441-2447 or 303-675-2140  
Fax: 303-675-2150  
[LDCForFreescaleSemiconductor@hibbertgroup.com](mailto:LDCForFreescaleSemiconductor@hibbertgroup.com)

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

For information on Freescale's Environmental Products program, go to <http://www.freescale.com/epp>.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.  
© Freescale Semiconductor, Inc. 2012. All rights reserved.