

# **MSP430 Interface to CC1100/2500 Code Library**

Keith Quiring

MSP430 Applications

## **ABSTRACT**

The MSP430 is an ideal microcontroller solution for low-cost, low-power wireless applications because it consumes very little power. The CC1100/CC2500 are market-leading RF devices in the ISM RF bands (Industrial, Scientific, and Medical). This library provides functions to facilitate the interfacing of an MSP430 device to these RF devices. Any device within the MSP430 family can be used with this library, made possible by hardware abstraction. Similarly, any SPI-capable interface module within the MSP430 family is supported by the library. This allows the designer maximum flexibility in choosing the best MSP430 device for the application. This document provides descriptive information and instructions for using the library either for demonstration purposes or implementation into a project.

## **Contents**

1	Introduction .....	1
2	Purpose and Scope .....	2
3	File Organization.....	2
4	Functions .....	3
5	Usage Considerations .....	4

## **List of Figures**

1	Code Library Stack .....	3
---	--------------------------	---

## **List of Tables**

1	Hardware Definition Files .....	2
2	Library Code.....	2
3	Application Included With the Library .....	2
4	Register Access Functions Provided by the Library .....	3

## **1 Introduction**

MSP430 is a great fit for any mobile application, where power conservation is a priority. The many power-saving mechanisms designed into the MSP430 make it ideal for such applications. An emerging mobile application is ISM-band (Industrial, Scientific, and Medical) and SRD-band (Short Range Device) wireless connections, in the 315/433/868/915-MHz and 2.4-GHz bands. Markets served by this application include AMR (Automatic Meter Reading), low-power telemetry, and wireless sensor networks.

Two market-leading devices that support this RF link are the CC1100/CC2500 from Chipcon (now part of TI). These are low-cost, single-chip UHF transceivers designed for very low-power wireless applications. The CC1100 operates up to 928 MHz, while the CC2500 operates at 2.4 GHz. One of these devices, paired with an MSP430 ultra-low-power microcontroller, forms a highly power-efficient wireless node that can transceive data at rates up to 500 kbps. The CC1100/CC2500 are each equipped with a SPI port, through which they can communicate with an MSP430.

## Purpose and Scope

This software is based on the *CC1100/CC2500 Examples and Libraries*, available from the TI product folder web pages for those devices.

## 2 Purpose and Scope

To aid in interfacing these devices, TI has produced a code library that eliminates the need to write low-level interface functions. It provides a boost in the development of an MSP430/CCxxxx-based product, saving time and allowing quick progression to the application-specific aspects of the project.

This library is designed to be used with any MSP430 device. Since a SPI master can be implemented using one of many peripherals within the MSP430 family, and since the peripherals available may differ by device and application, library calls are provided for each of these interfaces. The chosen interface is selected by assigning a value to a system variable, which causes the compiler to conditionally include the appropriate function calls. As such, application code utilizing the library remains portable between various MSP430 devices, with minimal modification required.

A complete example project is provided with the library. The purpose of this project is to demonstrate use of the library. It is not intended as a comprehensive guide to using the CC1100/CC2500, and it does not make use of all the features of these devices. It does, however, use all the register access functions provided by the library.

## 3 File Organization

The library has been implemented with modular hardware abstraction. There is a header file specific to each of the hardware components (CCxxxx, MSP430, and the board).

The hardware definition header files are shown in [Table 1](#). [Table 2](#) shows the library code file and its header, and [Table 3](#) shows the demonstration application that accompanies the library.

**Table 1. Hardware Definition Files**

Filename	Description
TI_CC_CC1100-CC2500.h	Definitions specific to the CC1100/2500 devices, including register locations and commonly-used masks for use with these registers.
TI_CC_MSP430.h	Definitions specific to the MSP430 device; primarily, the pins used in the SPI interface. Definitions for USART0/1, USCI_A0/1, USCI_B0/1, USI, and bit-banging are included. Also, labels are defined for use with the system variable RF_SER_INTF. This selects the modules to be used for the CCxxxx SPI interface.
TI_CC_hardware_board.h	Definitions specific to the board being used; that is, the connections between the MSP430 and CC1100/2500, such as the GDO pins. SPI connections are not defined here because they are defined inherently within TI_CC_MSP430.h.

**Table 2. Library Code**

Filename	Description
TI_CC_spi.c	Functions for accessing CC1100/CC2500 registers via SPI from MSP430.
TI_CC_spi.h	Function declarations for TI_CC_spi.c.

**Table 3. Application Included With the Library**

Filename	Description
CC1100-CC2500.c	Functions for programming the CC1100/CC2500, including calls for initialization, send, packet, and receive packet.
CC1100-CC2500.h	Function declarations for CC1100-CC2500.c.
include.h	High-level include file that lists all other include files.
main.c	Application code file.

Figure 1 shows a stack diagram of the library. Note that one of the files displayed in the stack is the standard definition file for the specific MSP430 device being used. This file is included with the development environment being used to create the MSP430 software.

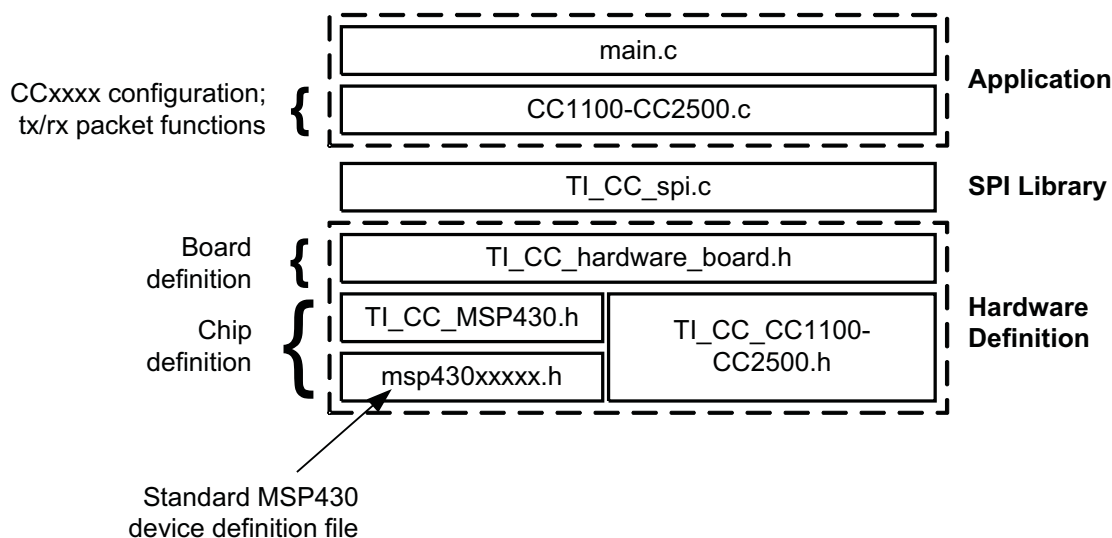


Figure 1. Code Library Stack

## 4 Functions

Table 4 shows the SPI register-access functions provided in the library, with a brief description.

Table 4. Register Access Functions Provided by the Library

Function Name	Description
<b>void</b> TI_CC_SPISetup( <b>void</b> )	Configures the SPI port assigned by the <i>RF_SER_INTF</i> system variable. Must be called before calling any of the other functions.
<b>void</b> TI_CC_power_up_reset_CCxxxx( <b>void</b> )	Implements the reset startup sequence recommended by the CC1100/CC2500 datasheet, including transmission of the <i>SRES</i> command strobe. Should be called after power to the CC1100/CC2500 is cycled.
<b>void</b> TI_CC_SPIWriteReg( <b>char</b> addr, <b>char</b> value)	Writes a byte <i>value</i> to the register at location <i>addr</i> .
<b>void</b> TI_CC_SPIWriteBurstReg( <b>char</b> addr, <b>char</b> *buffer, <b>char</b> count)	Writes values to multiple configuration registers, the first register being at address <i>addr</i> . The first data byte is at <i>buffer</i> , and both <i>addr</i> and <i>buffer</i> are incremented sequentially (within the CC1100/CC2500 and MSP430, respectively) until <i>count</i> writes have been performed.
<b>char</b> TI_CC_SPIReadReg( <b>char</b> addr)	Reads a single configuration register at address <i>addr</i> and returns the value read.
<b>void</b> TI_CC_SPIReadBurstReg( <b>char</b> addr, <b>char</b> *buffer, <b>char</b> count)	Reads multiple configuration registers, the first register being at address <i>addr</i> . The values read are deposited sequentially, starting at address <i>buffer</i> , until <i>count</i> registers have been read.
<b>char</b> TI_CC_SPIReadStatus( <b>char</b> addr)	Special read function for obtaining the value of status registers. Reads status register at address <i>addr</i> and returns the value read.
<b>void</b> TI_CC_SPIStrobe( <b>char</b> strobe)	Special write function for signaling command strobes. Writes to the strobe register located at address <i>addr</i> .

A version of these functions is provided for all MSP430 peripherals that are capable of communicating using the SPI protocol. These peripherals are:

- USART0
- USART1
- USCI\_A0
- USCI\_A1
- USCI\_B0
- USCI\_B1
- USI
- Bit-banging (emulation) with general I/O pins

## 5 Usage Considerations

### 5.1 Demo Project Hardware Configuration

The demo application is simple: pressing a switch on one board causes a corresponding LED on another board to toggle. The application supports up to four sets of LEDs and switches, located on pins identified within *TI\_CC\_hardware\_board.h*.

The demo application can be used with any of the standard carrier frequencies supported by CC1100/CC2500. The frequency is selected by the system variable *TI\_CC\_RF\_FREQ*, within *CC1100-CC2500.c*.

The configuration of the hardware definition files in the library as distributed by TI is for an MSP430F1612-equipped board with four LEDs and four switches. On this board, the SPI pins of the USART1 module are wired to the CC1100/CC2500. The system variable *TI\_CC\_RF\_SER\_INTF*, defined within *TI\_CC\_hardware\_board.h*, identifies this as the connected port, and therefore the compiler uses the code that supports this interface. The demo application makes use of the GDO0 output on the CC1100/CC2500, configuring it to output when a packet is being received.

Peripheral pinouts can change slightly between individual MSP430 devices and families. For this reason *TI\_CC\_MSP430.h* identifies the pins which correspond to a peripheral for any given device.

### 5.2 Adapting the Demo Project to Other Hardware

The procedure for adapting this code to other hardware is as follows:

- Edit the filename in the #include file reference at the top of *TI\_CC\_MSP430.h* (usually of format *msp430xxxx.h*), referencing a file from among the standard TI include files provided with the compiler, specific to the MSP430 device being used.
- Edit the pin assignments within *TI\_CC\_MSP430.h* for the interface modules being used. It is not necessary to modify the pins for the interfaces not selected for use with the SPI bus, as they will not be referenced by the library. The labels being referenced in the #define assignments will be drawn from the standard definition file listed at the top of *TI\_CC\_MSP430.h*.
- Edit the pin assignments in *TI\_CC\_hardware\_board.h*, taking into account all the necessary connections on the board being used. In the demo application, GDO1/2 are not used. The assigned labels are drawn from the standard definition file listed at the top of *TI\_CC\_MSP430.h*.
- Assign the proper values to *TI\_CC\_RF\_SER\_INTF* in *TI\_CC\_hardware\_board.h*. The labels available for assignment can be found at the bottom of *TI\_CC\_MSP430.h*.

After making these changes, rebuild the project and download the code image to two separate boards. The application should function as described earlier.

### 5.3 Using the Library With an Application

The same procedure as described in the section above should be applied in order to adapt the library to the new hardware. The switches and LEDs may not be used in the new application, but the chip select will be necessary, as may be the GDO signals.

The function `TI_CC_SPISetup()` should always be called after a POR event within the MSP430. The function `TI_CC_power_up_reset_CCxxxx` should always be called after a power-up event on the CC1100/CC2500.

After these calls, the access of registers is straightforward. The timing generated by the functions has been refined according to the CC1100/CC2500 datasheets, minimizing time spent performing the access in order to maximize power efficiency.

## IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

<b>Products</b>		<b>Applications</b>	
Amplifiers	<a href="http://amplifier.ti.com">amplifier.ti.com</a>	Audio	<a href="http://www.ti.com/audio">www.ti.com/audio</a>
Data Converters	<a href="http://dataconverter.ti.com">dataconverter.ti.com</a>	Automotive	<a href="http://www.ti.com/automotive">www.ti.com/automotive</a>
DSP	<a href="http://dsp.ti.com">dsp.ti.com</a>	Broadband	<a href="http://www.ti.com/broadband">www.ti.com/broadband</a>
Interface	<a href="http://interface.ti.com">interface.ti.com</a>	Digital Control	<a href="http://www.ti.com/digitalcontrol">www.ti.com/digitalcontrol</a>
Logic	<a href="http://logic.ti.com">logic.ti.com</a>	Military	<a href="http://www.ti.com/military">www.ti.com/military</a>
Power Mgmt	<a href="http://power.ti.com">power.ti.com</a>	Optical Networking	<a href="http://www.ti.com/opticalnetwork">www.ti.com/opticalnetwork</a>
Microcontrollers	<a href="http://microcontroller.ti.com">microcontroller.ti.com</a>	Security	<a href="http://www.ti.com/security">www.ti.com/security</a>
Low Power Wireless	<a href="http://www.ti.com/lpw">www.ti.com/lpw</a>	Telephony	<a href="http://www.ti.com/telephony">www.ti.com/telephony</a>
		Video & Imaging	<a href="http://www.ti.com/video">www.ti.com/video</a>
		Wireless	<a href="http://www.ti.com/wireless">www.ti.com/wireless</a>

Mailing Address: Texas Instruments  
Post Office Box 655303 Dallas, Texas 75265

Copyright © 2006, Texas Instruments Incorporated