CC85xx External Host Interface: Examples and Library

By Kjetil Holstad

Keywords

- *CC85xx*
- *Source Library*
- *External Host Interface*
- *MSP430*
- *EHIF*
- *CC85XXDK-HEADSET*
- *MSP430F5438*

1 Introduction

The CC85xx EHIF Library implements a generic framework for using the External Host Interface (EHIF), based on a 4-wire SPI interface and 2 optional GPIO pins. This allows external microcontroller or DSP, also referred to as the host processor, to implement flash programming and production tests, operate the CC85xx in host-controlled operation and check performance in autonomous operation:

- Autonomous operation
  - Device information
  - RF and audio statistics
  - Non-volatile storage read/write
- Host controlled operation
  - Device information
  - RF and audio statistics
  - Network management
  - Power management
  - Audio related operations
  - Data side-channel
  - Remote control functionality (mouse, keyboard, remote control)
  - Non-volatile storage read/write
- USB operation
  - Device information
  - RF and audio statistics
  - Non-volatile storage read/write
- Production test
  - RF tests (Rx, Tx, Packet Error Rate)
  - IO tests
  - Audio tests

This help system should not be read without also understanding the CC85XX Family User's Guide and especially its Appendix B.

The library code is documented in \ehif_lib\doc\cc85xx_ehif_lib.chm while the examples source code is documented extensively in the code itself. Project collateral discussed in this design note can be downloaded from the following URL: http://www.ti.com/lit/zip/SWRA369.

Note that for simple testing and extracting device information and statistics, the PurePath Wireless Commander PC tool may be favored over the EHIF Library. This tool supports scripted command execution (using Java-script), and uses the CCDebugger (included in the development kits) to interface with CC85xx devices.

Table of Contents

Abbreviations

API	Application Programming Interface
EHIF	External Host Inteface
DUT	Device Under Test
LCD	Liquid Crystal Display
PER	Packet Error Rate
SPI	Serial Peripheral Interface
USB	Universal Serial Bus

2 EHIF Library Source Code

The EHIF library source code is divided into the modules listed below. Click on the different modules' hyperlinks to view their documentation:

- Interfacing the host processor application:
  - **Command Execution Framework**
    - Used by the host processor to execute commands from the EHIF command set (except bootloader)
    - Application interface is based on the data structures defined by the **External Host Interface Definitions**
    - Automates endianess conversion when using little-endian microcontroller/compiler
  - **Basic Operations**
    - Implements basic SPI operations, pin and SPI-based reset routines and waiting/timeout utilities
  - **Bootloader Commands**
    - Provides access to bootloader specific EHIF commands and defines relevant EHIF status words
- Low-level microcontroller and board hardware abstraction:
  - **HAL: Board Specific Definitions and Routines**
    - Defines board specific constants, macros and functions.
  - **HAL: Microcontroller Specific Definitions and Routines**
    - Defines microcontroller specific constants, macros and functions.
- Internal functionality:
  - **Utilities**
  - **Basic Operations with Automatic Field Endianess Conversion**

2.1 Portability

The EHIF library and examples have been written and tested for MSP430 and 8051, using the IAR toolchain.
To support conversion from the big-endian nature of the EHIF commands to the microcontroller's and compiler's endianess, the **Basic Operations with Automatic Field Endianess Conversion** submodule (used by the **Command Execution Framework**) comes in two versions:

- The little-endian version swaps byte order for all 16- and 32-bit fields, using micro-scripts defined for each data structure
- The big-endian version is a simple byte-oriented wrapper around **Basic Operations**

Bit-fields in data structures have been used to maximize ease-of-use and application code readability while maintaining a low memory footprint. The understanding of bit-fields is compiler specific, and must be understood when selecting the appropriate version of the **External Host Interface Definitions** header file:

The little-endian version has all bit-fields within a container (e.g. uint32_t) reversed

- The big-endian version has all bit-fields in normal order as listed in the User's Guide

2.2 File Hierarchy

The library source code is found under the source directory:

- source: Endianess- and hardware-independent source files
  - little_endian: Little-endian versions of **External Host Interface Definitions** and **Basic Operations with Automatic Field Endianess Conversion**
  - big_endian: Big-endian versions of **External Host Interface Definitions** and **Basic Operations with Automatic Field Endianess Conversion**
  - hal: Low-level hardware abstraction

- *mcu*: One directory per MCU type, containing MCU-specific definitions
  - *board*: One directory per board type, containing board-specific definitions

3 Application Examples

Code examples are provided for the MSP-EXP430F5438 board, with project files and support libraries for IAR Embedded Workbench for MSP430 version 5.40.2. All applications utilize the board's LCD display for usage instructions and status output.

3.1 Flash Programming

Shows how to perform flash programming of CC85XX devices:

Uses the **Programming Algorithm** without modifications

- Two flash images have been converted from the hex-file format output by the PurePath Wireless Configurator to C byte-arrays, using the supplied HEX2C conversion tool. These CC85xx flash images are then compiled into the MSP430 flash image, so that the MSP430 can operate standalone:
  - Image 0 - Master network role from the CC85XXDK Preloaded Demo (autonomous operation), generated by running:

```
hex2c -p10 -ippweb_preloaded_demo_master.hex -oppweb_preloaded_demo_master.c -npMasterImage
```

  - Image 1 - Slave network role from the CC85XXDK Preloaded Demo (autonomous operation), generated by running:

```
hex2c -p10 -ippweb_preloaded_demo_slave.hex -oppweb_preloaded_demo_slave.c -npSlaveImage
```

3.2 Master

Shows how to operate a CC85xx host-controlled protocol master:

- Implements local CC85xx power control
- Implements button-based network pairing
- Implements remote volume control of the protocol slave

3.3 Slave

Shows how to operate a CC85xx host-controlled protocol slave:

- Implements local CC85xx power control
- Implements button-based network pairing
- Uses remote volume control
- Implements two different versions of the remote control user interface:
  - Play control: Volume up/down (affecting own volume through remote volume control) and play/pause and skip next/previous (for use with USB-based or other advanced protocol master)
    - Include *uif_play_control.c* in the build
  - Mouse (supported by FW 1.3.1 and later): Implements simple mouse functionality using the joystick (for use with USB-based protocol master)
    - Include *uif_mouse.c* in the build

3.4 Production Test

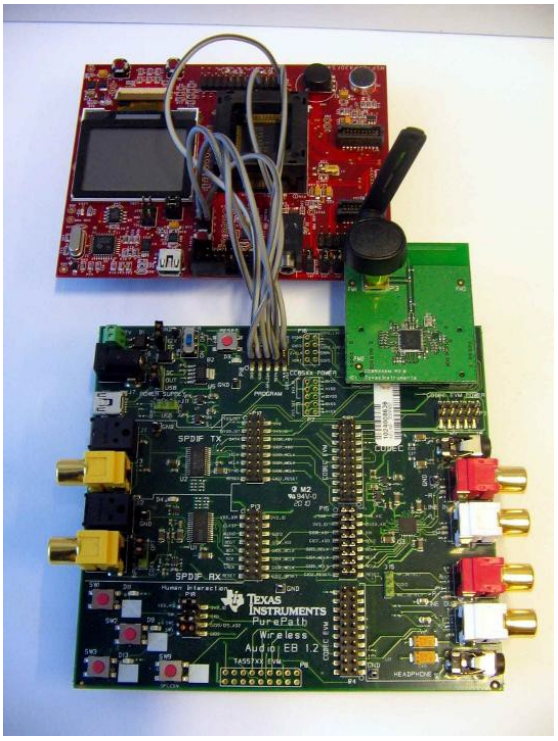
Shows how a production test can be implemented for the CC85xx Headset reference design.

- The test procedure checks I/O, audio and RF. Depending on customer priorities, a real production test can be more or less complex than this example
- A production test can also include flash programming, using the functionality demonstrated by the flash programming code example. When using the default MSP430 hardware configuration (with SPI at 4 MHz), the entire procedure (including erasing, programming and verification) can be completed approximately 350 ms.

4 Example Hardware Configuration

The example code is written for the MSP430F5438 Experimenter Board. Running an example requires one such board connected to a PurePath Wireless EB in the following manner: Note that the production test example with the CC85XXDK-HEADSET is connected in a similar fashion.

Function	PurePath Wireless EB	MSP-EXP430F5438
Ground	P10.1	GND
GIO3 (EHIF interrupt)	P10.4	P5.1
SPI CSn	P10.5	P10.3
SPI SCLK	P10.6	P10.0
RSTn	P10.7	P5.0
SPI MOSI	P10.8	P10.4
SPI MISO	P10.10	P10.5



5 References

[1] PurePath Wireless Headset Development Kit
http://focus.ti.com/docs/toolsw/folders/print/cc85xxdk-headset.html

[2] PurePath Wireless Configurator
http://www.ti.com/tool/purepath-wl-cfg

[3] CC Debugger Product Page
http://focus.ti.com/docs/toolsw/folders/print/cc-debugger.html

[4] MSP-EXP430F5438 Experimenter Board Product Page
http://focus.ti.com/docs/toolsw/folders/print/msp-exp430f5438.html

[5] SmartRF Flash Programmer Product Page
http://focus.ti.com/docs/toolsw/folders/print/flash-programmer.html

[6] MSP430 USB Debugging Interface Product Page
http://focus.ti.com/docs/toolsw/folders/print/msp-fet430uif.html

[7] CC85xx Family User's Guide
http://www.ti.com/lit/swru250

[8] PurePath Wireless Commander
http://www.ti.com/tool/purepath-wl-cmd

[9] PurePath Wireless Audio landing page
http://www.ti.com/purepathwireless

6 General Information

6.1 Document History

Revision	Date	Description/Changes
SWRA269A	2012.08.08	Renamed and completely re-written to include more examples and full EHIF source code compliant with FW1.4.0
SWRA369	2011.06.16	Initial release.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

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