UG103.8

EMBER® APPLICATION DEVELOPMENT FUNDAMENTALS: TOOLS

This document provides an overview of the toolchain used to develop, build, and deploy your applications, and discusses some additional tools and utilities.

New in This Revision

Initial release (contents previously published).

Contents

1 Introduction

As with most embedded development technologies, Silicon Labs provides a set of tools to allow you (the developer) to create a product using Ember ZigBee products. Each Ember chip family has a toolchain associated with it that addresses its unique development requirements. Wherever possible, we have selected the best development tools available, or we have developed the tool on our own.

This document provides an overview of the toolchain that you will use to develop, build and deploy your applications. These tools fall into one of three categories:

- EmberZNet PRO Stack Software
- Compiler Toolchain
- Application Development and Debugging Toolchain

The actual toolchain that you will use is device and processor model-dependent. For this discussion, processor model is either System-on-Chip (SoC) or Network Coprocessor (NCP). The SoC model requires that the customer application to be co-resident with the Ember stack. The NCP model requires that the customer application be on a separate host processor and the ZigBee stack run on the NCP. Table 1 summarizes the major tools for each device.

Table 1. Toolchain Summary

EmberZNet PRO Stack Software	Compiler	Application Development and Debugging
EM35x SoC		
Stack Libraries, HAL source, API Documentation, Sample Applications, Optional Development Kit	IAR EWARM: IDE: Compiler, Online Help; Debugger (device level); Document Library	Ember Desktop + Ember AppBuilder + Online Help
EM250 SoC		
Stack Libraries, HAL source, API Documentation, Sample Applications, Optional Development Kit	xIDE Compiler:IDE+Online Help; Debugger (device level); Document Library; Template Projects	Ember Desktop + Ember AppBuilder + Online Help
EM260 / EM35x NCP		
Stack Libraries, HAL source, API Documentation, Sample Applications, Utilities, Optional Development Kit	3rd Party Compiler Toolchain (depends upon Host Processor Selection)	Ember Desktop + Ember AppBuilder + Online Help

Misc Tools & Utilities

In addition to the major tools above, Silicon Labs also supplies a number of single function tools and utilities such as

- Bootloaders
- Programming Support Tools
- Token Utility

Silicon Labs also sells a variety of development kit hardware to suit various needs.

The following sections provide more detail about the most important elements of the toolchain.

2 EmberZNet Stack Software

The EmberZNet Stack Software is a collection of libraries, source code, tools, sample applications, and product documentation. The latest version of EmberZNet at the time of this writing is release 4.x, also known as EmberZNet PRO. Starting with EmberZNet 4.2, all Ember chip families are supported on the same stack release.

The EmberZNet 4.x is the Ember implementation of the ZigBee 2007 stack supporting the ZigBee PRO Feature set. ZigBee PRO supports mesh networks because of their increased flexibility and reliability. Consequently, all EmberZNet PRO applications must be linked with the stack library. Figure 1 illustrates how customer and EmberZNet PRO software interact.

In addition to this manual, additional resources are available for learning more about EmberZNet. These include:

- Document UG104, *Testing and Debugging Applications for Ember EM2xx and EM35x Platforms*
- Ember document UG105, *Advanced Application Programming with the Stack and HAL APIs*
- Platform-specific EmberZNet API References (HTML and PDF formats, provided with your installation)
- Ember sample applications (C files with many explanatory comments)
- Ember support portal www.silabs.com/zigbee-support

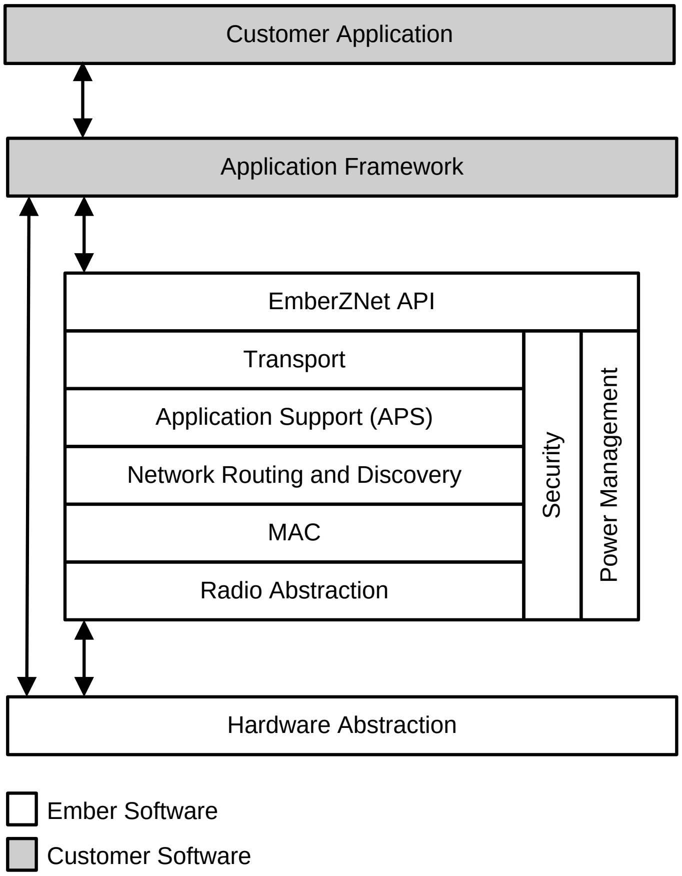

Figure 1. EmberZNet PRO and Customer Software Interaction

3 Compiler Toolchain

As mentioned above, the compiler toolchain is different based upon which Ember device and processor model you are using. The EM351/357 (EM35x) and EM250 devices, when being used as Systems-on-Chip, have customer-accessible processors. The EM35x use an ARM® Cortex -M3 processor and an IAR toolchain, while the EM250 uses a XAP core and the xIDE toolchain. Each of these toolchains provides similar capabilities that include a compiler, linker, debugger, sample applications, and user documentation.

Note: The xIDE toolchain is licensed through Silicon Labs, while the IAR toolchain is licensed through IAR. Specific information is supplied with your software.

The EM35x and EM260, when acting as a Network Co-Processor, are designed to be a coprocessor to whatever host you might be using. One of the most commonly selected hosts is an ARM processor, though many others can be used. You can get additional information about using the EM260, as well as the EM35x and EM250, on the Silicon Labs main website (www.silabs.com/zigbee-support) or at the Ember Support Portal (www.silabs.com/zigbee-support).

4 Ember Desktop

Ember Desktop helps you develop embedded applications through the Ember AppBuilder application development tool and the Network Analyzer utility. The Network Analyzer utility supports over-the-air debugging and testing of your products in a live wireless environment.

With Ember Desktop, you can monitor live networks and examine debug information provided by the network devices. Ember Desktop can be used with Ember development boards to set up an easy-to-use network for developing and testing an embedded application before deploying it on prototype or production hardware. Ember Desktop can also be used with custom hardware.

Ember AppBuilder is a rapid development tool that allows you to choose your application profile, such as Smart Energy or Home Automation, device type, cluster, and attribute set. Once you have defined your application through Ember AppBuilder, you can generate most of your application automatically through the use of Silicon Labs-supplied application code. This code has been through extensive testing, including ZigBee certification testing so it should save you significant development, test and certification time and expense.

For more detailed information about Ember AppBuilder and the Ember application framework, see document UG102, *Application Framework Developer Guide.*

5 Peripheral Drivers

Embedded source C code is provided for drivers of peripherals such as the serial controller and analog-to-digital converter (ADC). These drivers let you incorporate standard functionality into custom applications. For more information on these drivers, see the EmberZNet API reference for your platform.

6 Bootloaders

The bootloader is a program stored in reserved flash memory that allows a node to update its image on demand, either by serial communication or over the air. Production-level programming is typically done during the product manufacturing process, yet it is desirable to be able to reprogram the system after production is complete. More importantly, it is valuable to be able to update the device's firmware with new features and bug fixes after deployment. The bootloading capability makes that possible. See the document UG103.6, *Ember Application Development Fundamentals: Bootloaders* for more details.

7 Node Test

The nodetest applications provide low-level control of the radio and can be used to perform these tasks:

- Characterize radio performance.
- Set manufacturing and stack parameters (tokens).
- Verify proper functionality after manufacturing.
- Control the radio properly for the certification process required by many countries.

For more information about nodetest, see the relevant application note, listed in Table 2.

Table 2. Nodetest Application Notes

Document	**Device**
UG702, Bringing Up Custom Devices	EM250
UG704, Bringing Up Custom Devices	EM260
AN710, Bringing Up Custom Devices	EM35x

Most customers have standard product manufacturing test flows, but some do not incorporate RF testing. To address this issue, please see document AN700, *Manufacturing Test Guidelines*. This document describes the different options available for integrating RF testing and characterization into your standard test flows. This application note is intended for test engineers who are moving from the early prototype development stage to the manufacturing production environment and need assistance with manufacturing test process development. This application note applies to EM250, EM35x, and EM260 devices and stack releases EmberZNet 3.0 and later.

8 Utilities

8.1 Token Utility (for EM250)

The token utility application, available with source code in the top-level /hal directory, provides read and write access to non-volatile data (tokens) that are used by the EmberZNet PRO stack and application. You can use the utility to perform these tasks:

- View the memory map of the chip's non-volatile data storage based on the CONFIGURATION_HEADER and APPLICATION_TOKEN_HEADER used at compile time.
- View and set manufacturing data and stack parameters (tokens).
- View and set custom, non-volatile data used by the application (application tokens).
- Initialize the non-volatile data area for the chip.

For more about the token utility, see its description in /app/sampleApps.htm.

Note: EM35x does not offer a version of the Token Utility application or source. The nodetest application can be used for generic viewing of manufacturing and stack tokens as well as setting stack tokens. The nodetest application is not compatible with custom application tokens.

8.2 Hex File Utilities

A set of tools for manipulating hex files is also available. All of these utilities (em2xx_load, em2xx_read, em2xx_convert, em2xx_patch, em3xx_load, em3xx_convert, and em3xx_buildimage) are command line (DOS console) applications. The tools are listed in Table 3.

Table 3. Hex File Tools

Tool	Description
`em2xx_load/` `em3xx_load`	These utilities can be used to program the flash memory space of the EM250 through the SIF interface or the EM35x through the SWJ interface.

`em2xx_read or`
`em3xx_load`
	These utilities can be used to examine (dump) or generate a .hex file from the flash of the EM250 through SIF or the EM35x through SWJ.

`em2xx_convert`	This utility is intended for use in converting xIDE .xpv/.xdv application files into Ember .ebl bootload format or the Intel Hex format (.hex). In addition to the representation of the application, you can include a representation of the application bootloader of the customer manufacturing tokens.
`em3xx_convert`	This utility is intended for use in converting IAR .s37 application files into Ember .ebl bootload format. In addition to the representation of the application, you can include a representation of the application bootloader of the customer manufacturing tokens.
`em3xx_buildimage`	This utility is intended for use in manipulating EM35x file images, which includes generating Intel Hex format (.hex) files from a variety of sources, such as .s37 and .ebl files. In addition to the representation of the application, you can include a representation of the application bootloader of the customer manufacturing tokens.
`em2xx_patch`	This utility is used to program selected portions of the flash memory space of the EM2xx chips through the SIF interface. This utility can also be used to update specific portions of a .hex file
`em3xx_load`	This utility is used to program selected portions of the flash memory space of the EM35x chips through the SWJ interface.

Please refer to the Utilities Guide for the Ember chip you are using, either document UG106 for the EM2xx or document UG107 for the EM35x, for detailed information.

9 Development Programming Tools

The EM2xx USB Link programmer is a device programmer than can handle the programming requirements of the EM250 and EM260 devices in a development or small scale production environment. This programmer is USB 1.1 compatible and is fully compatible with the Ember EM250 and EM260 RCM modules and breakout boards. For more information about the EM2xx USB link, please contact customer support.

The Ember Debug Adapter (ISA3) provides the programming, debugging, and data emulation capability for an EM35x-based application. The Ember EM35x chip family integrates the ARM® Cortex™-M3, 32-bit microcontroller core. The Debug Adapter (ISA3) converts between the Ember JTAG and Serial Wire (SW) commands, Packet Trace Interface, TCP/IP, and UDP for an easy-to-deploy system over 10/100 Ethernet.

As part of the EM35x Development Kit, the Debug Adapter (ISA3) connects to the EM35X module through two interfaces: the 10-pin Packet Trace Port and the 12-pin data emulation interface (DEI). These two interfaces provide access to most EM35x GPIO as well as the EM35x programming and debug I/O. See document TS7, *Ember Debug Adapter (ISA3) Technical Specification*, for more information. See also documents AN707 and AN717, *Programming Options for the EM2xx* and *Programming Options for the EM35x*, respectively.

CONTACT INFORMATION

Silicon Laboratories Inc.

400 West Cesar Chavez
Austin, TX 78701
Tel: 1+(512) 416-8500
Fax: 1+(512) 416-9669
Toll Free: 1+(877) 444-3032

Please visit the Silicon Labs Technical Support web page for ZigBee products:
www.silabs.com/zigbee-support and register to submit a technical support request

Patent Notice
Silicon Labs invests in research and development to help our customers differentiate in the market with innovative low-power, small size, analog-intensive mixed-signal solutions. Silicon Labs' extensive patent portfolio is a testament to our unique approach and world-class engineering team.

The information in this document is believed to be accurate in all respects at the time of publication but is subject to change without notice. Silicon Laboratories assumes no responsibility for errors and omissions, and disclaims responsibility for any consequences resulting from the use of information included herein. Additionally, Silicon Laboratories assumes no responsibility for the functioning of undescribed features or parameters. Silicon Laboratories reserves the right to make changes without further notice. Silicon Laboratories makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Silicon Laboratories 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. Silicon Laboratories products are not designed, intended, or authorized for use in applications intended to support or sustain life, or for any other application in which the failure of the Silicon Laboratories product could create a situation where personal injury or death may occur. Should Buyer purchase or use Silicon Laboratories products for any such unintended or unauthorized application, Buyer shall indemnify and hold Silicon Laboratories harmless against all claims and damages.