MPLAB® 代码配置器的 LoRaWAN™ 库插件 用户指南

© 2017 Microchip Technology Inc.

请注意以下有关 Microchip 器件代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术指标。
- Microchip 确信：在正常使用的情况下， Microchip 系列产品是当今市场上同类产品中最安全的产品之一。
- 目前，仍存在着恶意、甚至是非法破坏代码保护功能的行为。就我们所知，所有这些行为都不是以 Microchip 数据手册中规定的操作规范来使用 Microchip 产品的。这样做的人极可能侵犯了知识产权。
- Microchip 愿与那些注重代码完整性的客户合作。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是 “牢不可破”的。

代码保护功能处于持续发展中。 Microchip 承诺将不断改进产品的代码保护功能。任何试图破坏 Microchip 代码保护功能的行为均可视为违反了 《数字器件千年版权法案 （Digital Millennium Copyright Act）》。如果这种行为导致他人在未经授权的情况下，能访问您的软件或其他受版权保护的成果，您有权依据该法案提起诉讼，从而制止这种行为。

提供本文档的中文版本仅为了便于理解。请勿忽视文档中包含的英文部分，因为其中提供了有关 Microchip 产品性能和使用情况的有用信息。Microchip Technology Inc. 及其分公司和相关公司、各级主管与员工及事务代理机构对译文中可能存在的任何差错不承担任何责任。建议参考 Microchip Technology Inc. 的英文原版文档。

本出版物中所述的器件应用信息及其他类似内容仅为您提供便利，它们可能由更新之信息所替代。确保应用符合技术规范，是您自身应负的责任。**Microchip 对这些信息不作任何明示或暗示、书面或口头、法定或其他形式的声明或担保，包括但不限于针对其使用情况、质量、性能、适销性或特定用途的适用性的声明或担保。** Microchip 对因这些信息及使用这些信息而引起的后果不承担任何责任。如果将 Microchip 器件用于生命维持和 / 或生命安全应用，一切风险由买方自负。买方同意在由此引发任何一切伤害、索赔、诉讼或费用时，会维护和保障 Microchip 免于承担法律责任，并加以赔偿。除非另外声明，在 Microchip 知识产权保护下，不得暗中或以其他方式转让任何许可证。

Microchip 位于美国亚利桑那州 Chandler 和 Tempe 与位于俄勒冈州 Gresham的全球总部、设计和晶圆生产厂及位于美国加利福尼亚州和印度的设计中心均通过了 ISO/TS-16949:2009 认证。Microchip 的 PIC® MCU 与 dsPIC® DSC、KEELOQ® 跳码器件、串行 EEPROM、单片机外设、非易失性存储器和模拟产品严格遵守公司的质量体系流程。此外，Microchip 在开发系统的设计和生产方面的质量体系也已通过了 ISO 9001:2000 认证。

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
═ ISO/TS 16949 ═

商标

Microchip 的名称和徽标组合、 Microchip 徽标、 AnyRate、 AVR、 AVR 徽标、 AVR Freaks、 BeaconThings、 BitCloud、 CryptoMemory、 CryptoRF、 dsPIC、 FlashFlex、 flexPWR、 Heldo、 JukeBlox、 KEELOQ、 KEELOQ 徽标、 Kleer、 LANCheck、 LINK MD、 maXStylus、 maXTouch、 MediaLB、 megaAVR、 MOST、 MOST 徽标、 MPLAB、 OptoLyzer、 PIC、 picoPower、 PICSTART、 PIC32 徽标、 Prochip Designer、 QTouch、 RightTouch、 SAM-BA、 SpyNIC、 SST、 SST 徽标、 SuperFlash、 tinyAVR、 UNI/O 及 XMEGA 均为 Microchip Technology Inc. 在美国和其他国家或地区的注册商标。

ClockWorks、 The Embedded Control Solutions Company、 EtherSynch、 Hyper Speed Control、 HyperLight Load、 IntelliMOS、 mTouch、 Precision Edge 和 Quiet-Wire 均为 Microchip Technology Inc. 在美国的注册商标。

Adjacent Key Suppression、 AKS、 Analog-for-the-Digital Age、 Any Capacitor、 AnyIn、 AnyOut、 BodyCom、 chipKIT、 chipKIT 徽标、 CodeGuard、 CryptoAuthentication、 CryptoCompanion、 CryptoController、 dsPICDEM、 dsPICDEM.net、 Dynamic Average Matching、 DAM、 ECAN、 EtherGREEN、 In-Circuit Serial Programming、 ICSP、 Inter-Chip Connectivity、 JitterBlocker、 KleerNet、 KleerNet 徽标、 Mindi、 MiWi、 motorBench、 MPASM、 MPF、 MPLAB Certified 徽标、 MPLIB、 MPLINK、 MultiTRAK、 NetDetach、 Omniscient Code Generation、 PICDEM、 PICDEM.net、 PICkit、 PICtail、 PureSilicon、 QMatrix、 RightTouch 徽标、 REAL ICE、 Ripple Blocker、 SAM-ICE、 Serial Quad I/O、 SMART-I.S.、 SQI、 SuperSwitcher、 SuperSwitcher II、 Total Endurance、 TSHARC、 USBCheck、 VariSense、 ViewSpan、 WiperLock、 Wireless DNA 和 ZENA 均为 Microchip Technology Inc. 在美国和其他国家或地区的商标。

SQTP 为 Microchip Technology Inc. 在美国的服务标记。

Silicon Storage Technology 为 Microchip Technology Inc. 在除美国外的国家或地区的注册商标。

GestIC 为 Microchip Technology Inc. 的子公司 Microchip Technology Germany II GmbH & Co. & KG 在除美国外的国家或地区的注册商标。

在此提及的所有其他商标均为各持有公司所有。

© 2017, Microchip Technology Inc. 版权所有。

ISBN：978-1-5224-1508-4

声明对象：MPLAB®代码配置器的LoRaWAN™库插件

EU Declaration of Conformity

This declaration of conformity is issued by the manufacturer.
The development/evaluation tool is designed to be used for research and development in a laboratory environment. This development/evaluation tool is not a Finished Appliance, nor is it intended for incorporation into Finished Appliances that are made commercially available as single functional units to end users under EU EMC Directive 2004/108/EC and as supported by the European Commission's Guide for the EMC Directive 2004/108/EC (8th February 2010).
This development/evaluation tool complies with EU RoHS2 Directive 2011/65/EU.
This development/evaluation tool, when incorporating wireless and radio-telecom functionality, is in compliance with the essential requirement and other relevant provisions of the R&TTE Directive 1999/5/EC and the FCC rules as stated in the declaration of conformity provided in the module datasheet and the module product page available at www.microchip.com.
For information regarding the exclusive, limited warranties applicable to Microchip products, please see Microchip's standard terms and conditions of sale, which are printed on our sales documentation and available at www.microchip.com.
Signed for and on behalf of Microchip Technology Inc. at Chandler, Arizona, USA.

Derek Carlson
VP Development Tools

11-NOV-16
Date

注：

© 2017 Microchip Technology Inc.

目录

前言 9

第1章 概述

1.1 简介 13
1.2 向MCC添加LoRaWAN库插件 14

第2章 LoRaWAN™ 库插件

2.1 LoRaWAN基本配置 17
2.1.1 LoRaWAN简单设置 18
2.1.1.1 定时器选择 18
2.1.1.2 无线电模块选择 19
2.1.1.3 SPI模块选择 19
2.1.1.4 ISM频段选择 20
2.1.1.5 默认LoRaWAN类别选择 20
2.1.2 LoRaWAN高级设置 21
2.1.2.1 欧洲433 MHz高级设置 21
2.1.2.2 欧洲868 MHz高级设置 26
2.1.2.3 北美915 MHz高级设置 30

第3章 运行在RN2XX3模块上的LoRaWAN™ 库插件

3.1 概述 33
3.2 RN2XX3模块上的LoRaWAN配置 33
3.2.1 LoRaWAN项目创建 33
3.2.2 LoRaWAN基本配置 37
3.2.3 LoRaWAN RN2XX3配置 40
3.2.3.1 概述 40
3.2.3.2 向项目添加需要的资源 41
3.2.3.3 LoRaWAN RN2903系统模块配置 43
3.2.3.4 LoRaWAN RN2903 TMR1模块配置 44
3.2.3.5 LoRaWAN RN2903 MSSP2模块配置 45
3.2.3.6 LoRaWAN RN2903 DIO引脚配置 46
3.2.3.7 LoRaWAN RN2903引脚模块配置 47
3.3 RN2XX3模块的LoRaWAN™ 协议栈生成 49

第4章 LoRaWAN™ 协议栈API

4.1 概述 51
4.2 LoRaWAN架构 51
4.2.1 LoRaWAN项目创建 51
4.2.2 文件结构 52
4.3 应用程序编程接口 54
4.3.1 概述 54
4.3.2 LoRaWAN API 54
4.3.2.1 LORAWAN_Init 54
4.3.2.2 LORAWAN_Join 54

© 2017 Microchip Technology Inc.

4.3.2.3 LORAWAN_Send 55
4.3.2.4 LORAWAN_SetDeviceEui 55
4.3.2.5 LORAWAN_GetDeviceEui 55
4.3.2.6 LORAWAN_SetApplicationEui 56
4.3.2.7 LORAWAN_GetApplicationEui 56
4.3.2.8 LORAWAN_SetDeviceAddress 56
4.3.2.9 LORAWAN_GetDeviceAddress 57
4.3.2.10 LORAWAN_SetNetworkSessionKey 57
4.3.2.11 LORAWAN_GetNetworkSessionKey 58
4.3.2.12 LORAWAN_SetApplicationSessionKey 58
4.3.2.13 LORAWAN_GetApplicationSessionKey 59
4.3.2.14 LORAWAN_SetApplicationKey 59
4.3.2.15 LORAWAN_GetApplicationKey 60
4.3.2.16 LORAWAN_SetAdr 60
4.3.2.17 LORAWAN_GetAdr 61
4.3.2.18 LORAWAN_GetCurrentDataRate 61
4.3.2.19 LORAWAN_SetTxPower 62
4.3.2.20 LORAWAN_GetTxPower 62
4.3.2.21 LORAWAN_SetSyncWord 62
4.3.2.22 LORAWAN_GetSyncWord 63
4.3.2.23 LORAWAN_SetUplinkCounter 63
4.3.2.24 LORAWAN_GetUplinkCounter 63
4.3.2.25 LORAWAN_SetDownlinkCounter 64
4.3.2.26 LORAWAN_GetDownlinkCounter 64
4.3.2.27 LORAWAN_SetReceiveDelay1 64
4.3.2.28 LORAWAN_GetReceiveDelay1 65
4.3.2.29 LORAWAN_GetReceiveDelay2 65
4.3.2.30 LORAWAN_SetJoinAcceptDelay1 66
4.3.2.31 LORAWAN_GetJoinAcceptDelay1 66
4.3.2.32 LORAWAN_SetJoinAcceptDelay2 67
4.3.2.33 LORAWAN_GetJoinAcceptDelay2 67
4.3.2.34 LORAWAN_SetMaxFcntGap 68
4.3.2.35 LORAWAN_GetMaxFcntGap 68
4.3.2.36 LORAWAN_SetAdrAckLimit 69
4.3.2.37 LORAWAN_GetAdrAckLimit 69
4.3.2.38 LORAWAN_SetAdrAckDelay 70
4.3.2.39 LORAWAN_GetAdrAckDelay 70
4.3.2.40 LORAWAN_SetAckTimeout 71
4.3.2.41 LORAWAN_GetAckTimeout 71
4.3.2.42 LORAWAN_SetNumberOfRetransmissions 71
4.3.2.43 LORAWAN_GetNumberOfRetransmissions 72
4.3.2.44 LORAWAN_SetReceiveWindow2Parameters 72
4.3.2.45 LORAWAN_GetReceiveWindow2Parameters 73
4.3.2.46 LORAWAN_SetBattery 73
4.3.2.47 LORAWAN_GetPrescaler 74
4.3.2.48 LORAWAN_SetAutomaticReply 74
4.3.2.49 LORAWAN_GetAutomaticReply 75
4.3.2.50 LORAWAN_GetStatus 75
4.3.2.51 LORAWAN_GetLinkCheckMargin 76
4.3.2.52 LORAWAN_GetLinkCheckGwCnt 76
4.3.2.53 LORAWAN_GetFrequency 77

© 2017 Microchip Technology Inc.

4.3.2.54 LORAWAN_SetDataRange 77
4.3.2.55 LORAWAN_GetDataRange 78
4.3.2.56 LORAWAN_SetChannelIdStatus 78
4.3.2.57 LORAWAN_GetChannelIdStatus 78
4.3.2.58 LORAWAN_Pause 79
4.3.2.59 LORAWAN_Resume 79
4.3.2.60 LORAWAN_LinkCheckConfigure 80
4.3.2.61 LORAWAN_ForceEnable 80
4.3.2.62 LORAWAN_Reset（用于EU433/EU868 ISM频段） 80
4.3.2.63 LORAWAN_Reset（用于NA915 ISM频段） 81
4.3.2.64 LORAWAN_SetFrequency（仅用于EU433/EU868） 81
4.3.2.65 LORAWAN_SetDutyCycle（仅用于EU433/EU868） 82
4.3.2.66 LORAWAN_GetDutyCycle（仅用于EU433/EU868） 82
4.3.2.67 LORAWAN_GetISMBand（仅用于EU433/EU868） 83
4.3.2.68 LORAWAN_Mainloop 83
4.3.2.69 LORAWAN_SetClass 83
4.3.2.70 LORAWAN_GetClass 84
4.3.2.71 LORAWAN_SetMcast 84
4.3.2.72 LORAWAN_GetMcast 84
4.3.2.73 LORAWAN_SetMcastApplicationSessionKey 85
4.3.2.74 LORAWAN_GetMcastApplicationSessionKey 85
4.3.2.75 LORAWAN_SetMcastNetworkSessionKey 86
4.3.2.76 LORAWAN_GetMcastNetworkSessionKey 86
4.3.2.77 LORAWAN_SetMcastDeviceAddress 87
4.3.2.78 LORAWAN_GetMcastDeviceAddress 87
4.3.2.79 LORAWAN_SetMcastDownCounter 87
4.3.2.80 LORAWAN_GetMcastDownCounter 88
4.3.2.81 LORAWAN_GetState 88

第5章 构建基于LoRaWAN™的应用程序

5.1 概述 89
5.2 基本LoRaWAN A类协议栈操作 89
5.2.1 概述 89
5.2.2 初始化 89
5.2.2.1 有效负载接收指示（有效负载接收回调） 90
5.2.2.2 激活指示响应 91
5.2.3 激活配置 91
5.2.3.1 个性化激活配置 91
5.2.3.2 无线激活配置 93
5.2.4 激活 94
5.2.4.1 个性化激活 94
5.2.4.2 无线激活 94
5.2.5 通信 94
5.3 基本LoRaWAN C类协议栈操作 95
5.3.1 概述 95
5.3.2 类别更改 95
5.3.3 C类连续接收 96
5.3.4 C类多播消息 96
5.4 构建基于LoRaWAN的自定义应用程序 96
5.4.1 概述 96
5.4.2 中断允许 96
5.4.3 LoRaWAN Mainloop函数 96

5.5 LoRaWAN协议栈加密 97
5.6 基于LoRaWAN的自定义应用程序示例 98
全球销售及服务网点 101

© 2017 Microchip Technology Inc.

前言

客户须知

所有文档均会过时，本文档也不例外。 **Microchip** 的工具和文档将不断演变以满足客户的需求，因此实际使用中有些对话框和 / 或工具说明可能与本文档所述之内容有所不同。请访问我们的网站（**www.microchip.com**）获取最新文档。

文档均标记有 **“DS”** 编号。该编号出现在每页底部的页码之前。 **DS** 编号的命名约定为 **“DSXXXXXXXXA_CN”**，其中 **“XXXXXXXX”** 为文档编号，**“A”** 为文档版本。

欲了解开发工具的最新信息，请参考 **MPLAB® IDE** 在线帮助。从 **Help** （帮助）菜单选择 **Topics** （主题），打开现有在线帮助文件列表。

简介

本章包含使用MCC LoRaWAN™ 库插件前需了解的一般信息。内容包括：

- 文档编排
- 本指南使用的约定
- 推荐读物
- Microchip网站
- 开发系统变更通知客户服务
- 客户支持
- 版本历史

文档编排

本文档介绍了如何使用MCC LoRaWAN库插件作为开发工具在目标板上仿真和调试固件。文档内容编排如下：

- **第1章“概述”**——向用户介绍用于MPLAB代码配置的LoRaWAN库插件并概述库功能。
- **第2章“LoRaWAN™ 库插件”**——介绍LoRaWAN库的基本配置和高级配置。
- **第3章“运行在RN2XX3模块上的LoRaWAN™ 库插件”**——详细介绍RN2XX3模块上的LoRaWAN配置。
- **第4章“LoRaWAN™ 协议栈API”**——介绍LoRaWAN协议栈为用户提供的API。
- **第5章“构建基于LoRaWAN™ 的应用程序”**——介绍LoRaWAN的基本工作原理并提供基于LoRaWAN的自定义应用程序的示例。

本指南使用的约定

本指南采用以下文档约定：

文档约定

说明	表示	示例
Arial 字体：		
斜体字	参考书目	*MPLAB® IDE User's Guide*
	需强调的文字	…… 为*仅有*的编译器 ……
首字母大写	窗口	Output 窗口
	对话框	Settings 对话框
	菜单选择	选择 Enable Programmer
引用	窗口或对话框中的字段名	“Save project before build”
带右尖括号且有下划线的斜体文字	菜单路径	*File>Save*
粗体字	对话框按钮	单击 **OK**
	选项卡	单击 **Power** 选项卡
N‘Rnnnn	verilog 格式的数字，其中 N 为总位数， R 为基数， n 为其中一位。	4‘b0010， 2‘hF1
尖括号 < > 括起的文字	键盘上的按键	按下 <Enter>，<F1>
Courier New 字体：		
常规 Courier New	源代码示例	`#define START`
	文件名	`autoexec.bat`
	文件路径	`c:\mcc18\h`
	关键字	`_asm`， `_endasm`， `static`
	命令行选项	`-Opa+`，`-Opa-`
	二进制位值	`0`， `1`
	常量	`0xFF`， `'A'`
斜体 Courier New	可变参数	*`file`*`.o`，其中 *`file`* 可以是任一有效文件名
方括号 [ ]	可选参数	`mcc18 [选项] file [选项]`
花括号和竖线：{ | }	选择互斥参数； “或”选择	`errorlevel {0|1}`
省略号 ...	代替重复文字	`var_name [, var_name...]`
	表示由用户提供的代码	`void main (void) { ... }`

© 2017 Microchip Technology Inc.

推荐读物

本用户指南介绍了如何使用MCC LoRaWAN库插件。有关使用其他工具的最新信息，请参见MPLAB® X IDE主页：www.microchip.com/mplabx/。该资源页包含其他MPLAB X兼容工具和插件等的更新文档、下载和链接。

其他推荐读物为LoRaWAN规范1.0。该规范介绍了LoRaWAN协议，由LoRa®联盟在http://www.lora-alliance.org上提供。

MICROCHIP网站

Microchip 网站 （www.microchip.com）为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的互联网浏览器即可访问，网站提供以下信息：

- **产品支持** —— 数据手册和勘误表、应用笔记和示例程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及归档软件
- **一般技术支持** —— 常见问题解答 （FAQ）、技术支持请求、在线讨论组以及 Microchip 顾问计划成员名单
- **Microchip 业务** —— 产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排表、 Microchip 销售办事处、代理商以及工厂代表列表

开发系统变更通知客户服务

Microchip 的客户通知服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

要注册，请先访问 Microchip 网站 www.microchip.com，点击 “变更通知客户” （Customer Change Notification），然后按照注册指示完成注册。

开发系统产品的分类如下：

- **编译器** ——Microchip C 编译器及其他语言工具的最新信息，包括 MPLAB C 编译器、MPASM™ 和 MPLAB 16 位汇编器、MPLINK™ 和 MPLAB 16 位目标链接器，以及 MPLIB™ 和 MPLAB 16 位目标库管理器。
- **仿真器** ——Microchip MPLAB REAL ICE™ 在线仿真器的最新信息。
- **在线调试器** ——Microchip 在线调试器的最新信息，包括 MPLAB ICD 3 和 PICkit™ 3。
- **MPLAB IDE**—— 关于开发系统工具的 Windows® 集成开发环境 Microchip MPLAB IDE 的最新信息，主要针对 MPLAB IDE、 MPLAB SIM 软件模拟器、 MPLAB IDE 项目管理器，以及一般编辑和调试功能。
- **编程器** ——Microchip 编程器的最新信息，包括 MPLAB PM3 器件编程器和 PICkit 3 开发编程器。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师 （FAE）
- 技术支持

客户应联系其代理商、代表或应用工程师 （FAE）寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过 http://microchip.com/support 获得网上技术支持。

版本历史

版本A（2016年11月）

本文档的初始版本。

版本B（2017年1月）

对第2章和第3章进行了少量更新。

在第4章“LoRaWAN™ 协议栈API”中增加了新功能。

增加了第5.3节“基本LoRaWAN C类协议栈操作”。

© 2017 Microchip Technology Inc.

第1章 概述

1.1 简介

凭借MPLAB®代码配置器的LoRaWAN™库插件，可以轻松快速地为面向LoRa®技术终端设备的Microchip LoRaWAN协议栈解决方案生成C代码。

LoRaWAN库插件以Java归档文件的形式提供（文件扩展名为`.jar`），必须添加到MPLAB代码配置器中。

LoRaWAN协议栈目前仅支持8位PIC®器件。PIC器件运行LoRaWAN协议栈需满足的最低要求如下：

- 32 KB闪存
- 3 KB RAM存储器
- 1个SPI
- 6个GPIO（其中三个GPIO必须支持中断功能）。如果PIC器件支持外设引脚选择功能，则需要使用一个额外的GPIO来实现SPI通信的片选功能。

该库插件使用图形用户界面（Graphical User Interface，GUI）来实现以下目的：

- 根据器件资源（器件上的模块）选择LoRaWAN协议栈的要求
- 针对EU 433/868 MHz ISM频段定义通信通道
- 针对NA 915 MHz ISM频段使能/禁止通信通道
- 针对与服务器之间的通信调整LoRaWAN协议栈的各个参数
- 生成编程PIC®单片机所需的C代码

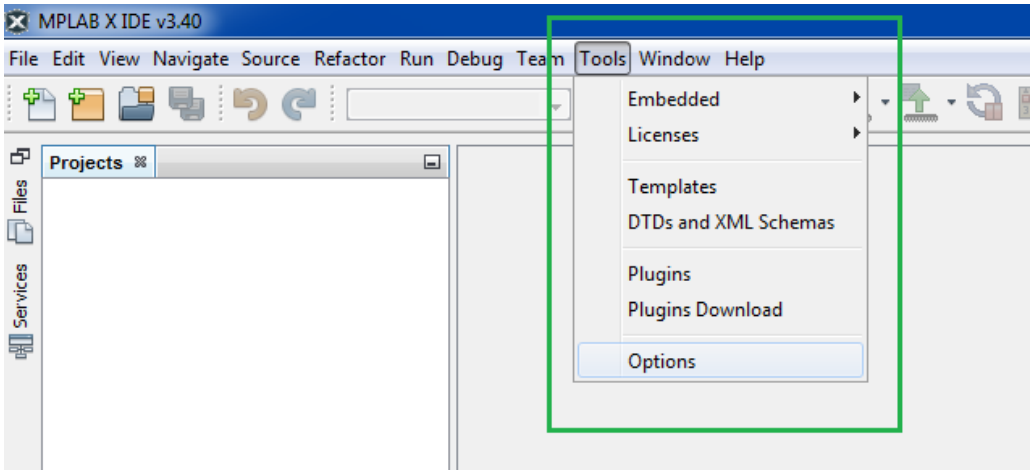
本用户指南的后续章节涵盖LoRaWAN库插件的各个组件，并介绍设置基本LoRaWAN项目的具体步骤。如果需要任何其他信息或有任何疑问，请联系您当地的LoRaWAN代表。

© 2017 Microchip Technology Inc.

1.2 向MCC添加LoRaWAN库插件

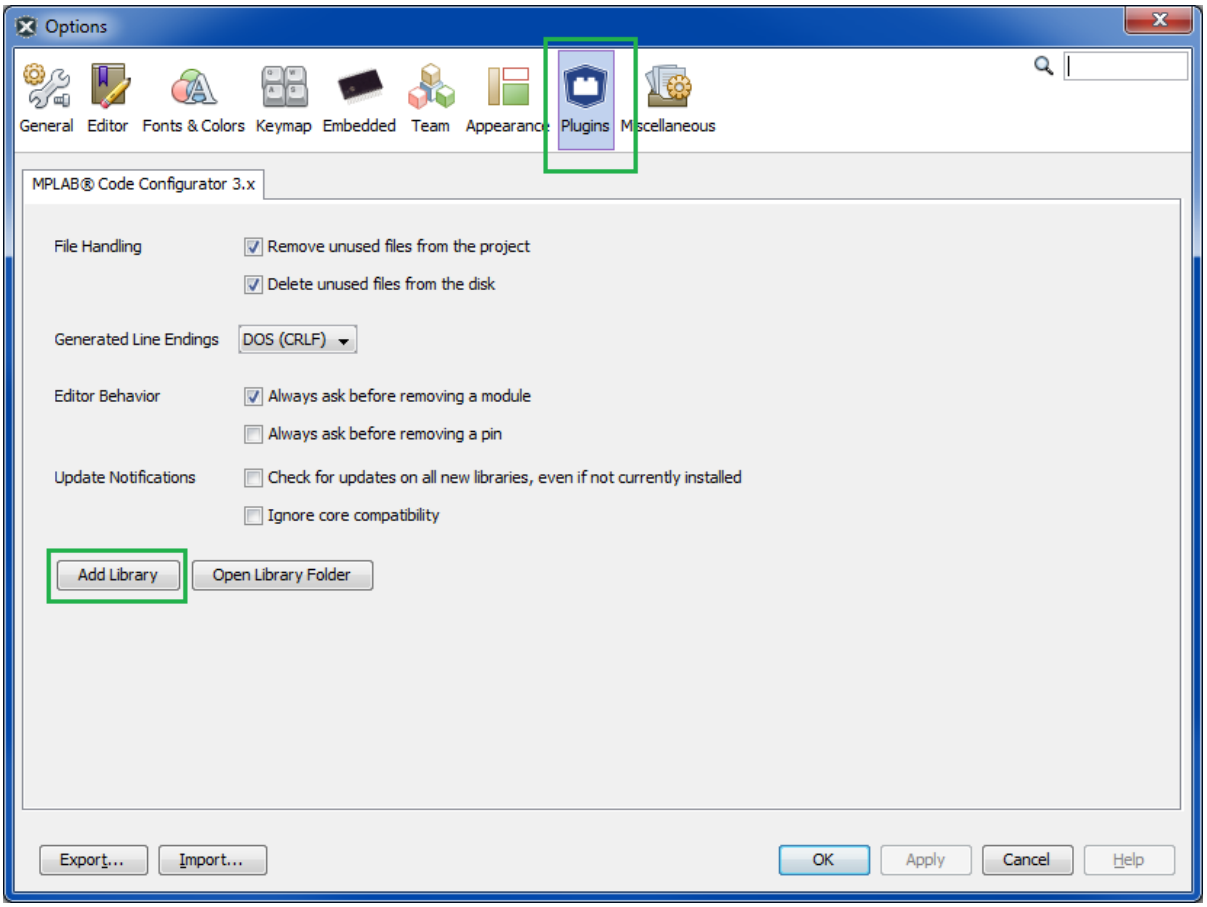
要添加LoRaWAN库插件，用户必须打开MPLAB X IDE，然后单击**Tools -> Options**（工具 -> 选项）（见图1-1）。Options窗口随即打开。

图1-1： 向MCC添加LoRaWAN™库插件



在Options窗口中，单击**Plug-ins**（插件）选项卡，然后按下**Add Library**（添加库）按钮（见图1-2）。

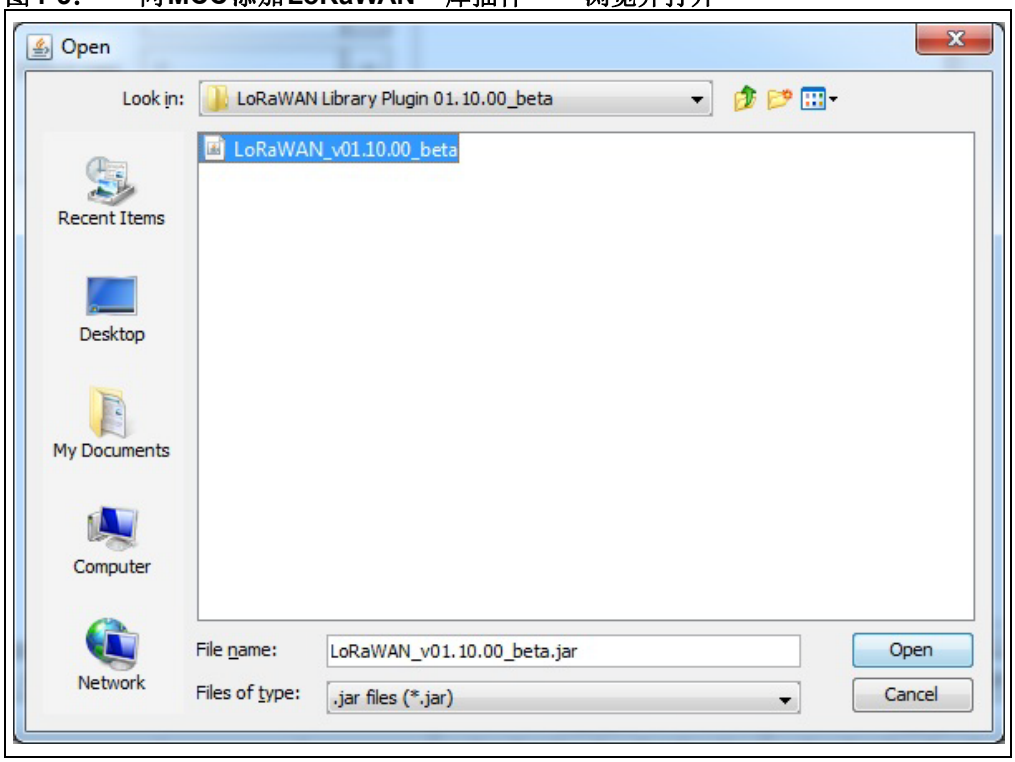
图1-2： 向MCC添加LoRaWAN™库插件——OPTIONS窗口



© 2017 Microchip Technology Inc.

浏览至LoRaWAN库插件所在的位置，选中该插件，然后单击**Open**（打开），将其添加到MCC中（见图1-3）。

图1-3：　向MCC添加LoRaWAN™库插件——浏览并打开



注：	如果LoRaWAN™库插件已添加到MCC中，将弹出一条消息，询问用户是否覆盖原有文件。

注：

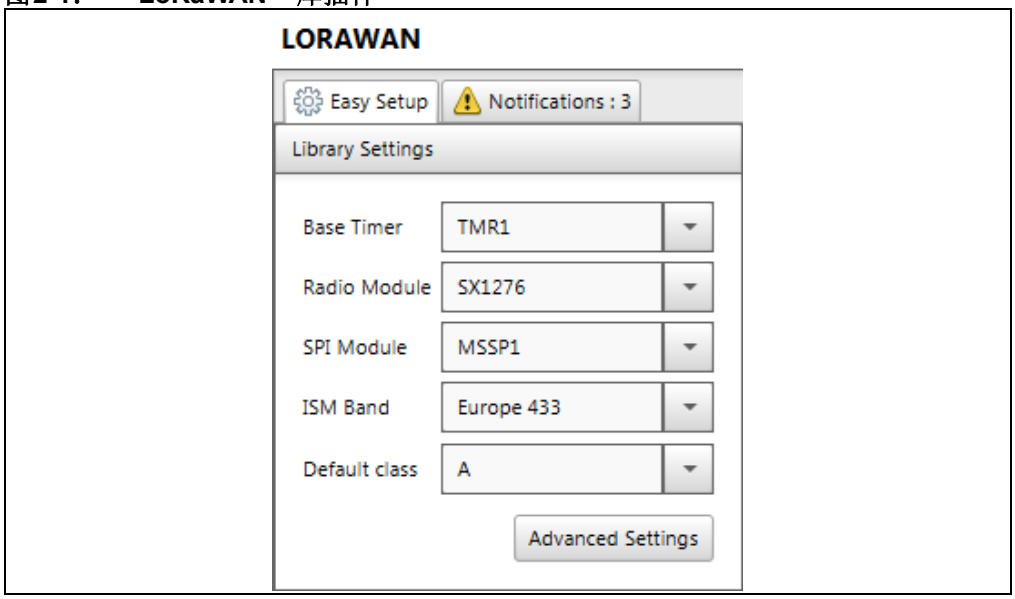
© 2017 Microchip Technology Inc.

第2章 LoRaWAN™库插件

LoRaWAN库插件具有向导菜单，如下面的图2-1所示：

- 它提供针对基本用户的最低设置要求
- 它还提供Advanced Settings（高级设置）选项以访问LoRaWAN协议栈的更多设置

图2-1： LoRaWAN™库插件



2.1 LoRaWAN基本配置

用户必须设置以下五个基本选项才能使LoRaWAN协议栈正常工作。这些选项必须反映所使用的LoRaWAN终端设备硬件（例如，无线电收发器和单片机的MSSP引脚等）：

1. Base Timer（基本定时器）
2. Radio Module（无线电模块）
3. SPI Module（SPI模块）
4. ISM Band（ISM频段）
5. Default Class（默认类别）

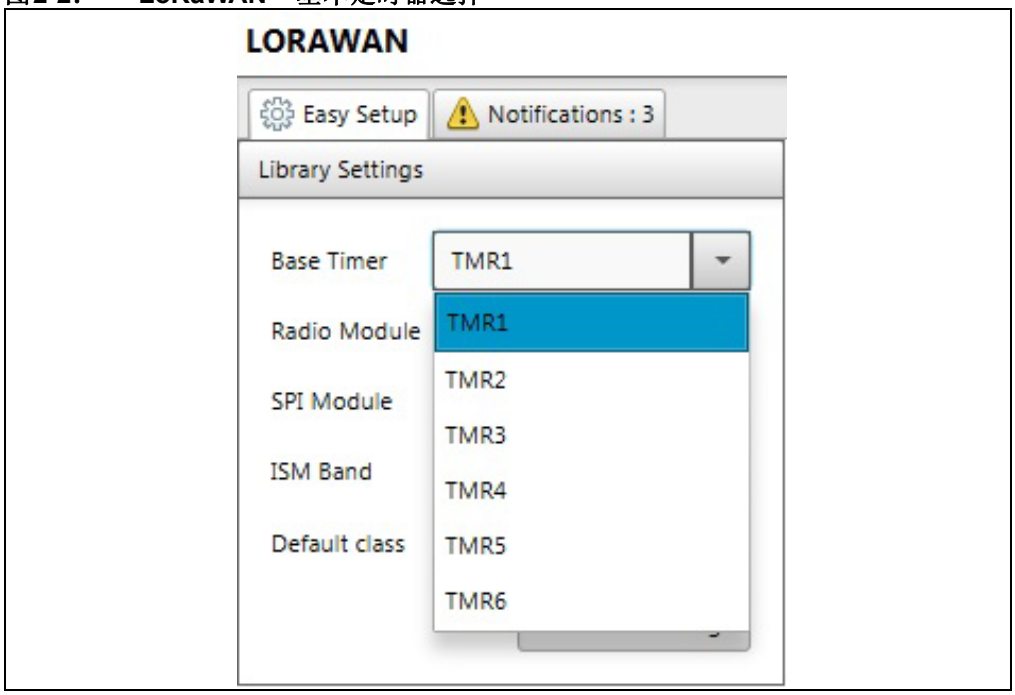
该库插件将根据单片机资源、支持的无线电模块或LoRaWAN协议栈的使用地区为上述设置生成选项列表以供选择。

2.1.1 LoRaWAN简单设置

2.1.1.1 定时器选择

必须从单片机提供的定时器列表中选择将供LoRaWAN协议栈使用的定时器，如图2-2所示。用户必须进行检查，确保选择16位定时器，因为这是LoRaWAN库的特定要求。

图2-2： LoRaWAN™基本定时器选择



> **注：** LoRaWAN协议栈目前仅支持16位定时器。定时器时钟源必须为外部时钟源，在32.768 kHz下运行。更多详细信息，请参见**第3.2.3.4节“LoRaWAN RN2903 TMR1模块配置”**中的LoRaWAN协议栈定时器配置。

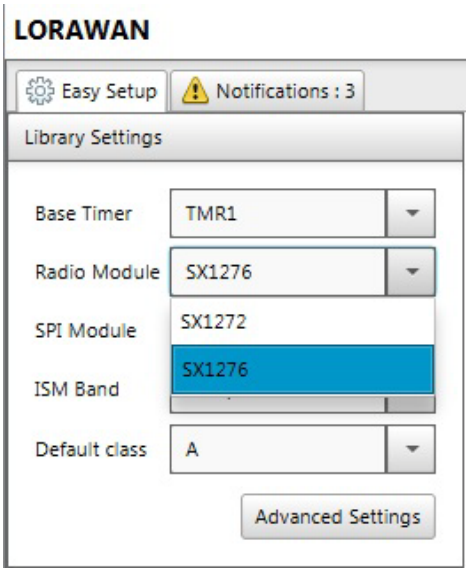
© 2017 Microchip Technology Inc.

2.1.1.2 无线电模块选择

目前，LoRaWAN协议栈仅支持两款无线电收发器：Semtech SX1272和Semtech SX1276。用户必须选择终端设备上使用的那款无线电收发器，如图2-3所示。

> **注：** SX1272不支持433 MHz频段。

图2-3： LoRaWAN™无线电模块选择

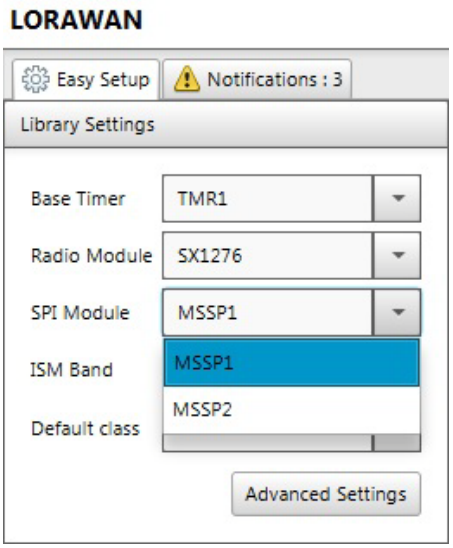


2.1.1.3 SPI模块选择

终端设备由一个PIC器件和一个无线电收发器组成。二者之间需要进行SPI通信，其中PIC器件为主器件，而无线电收发器为从器件。用户必须选择在PIC单片机和无线电收发器之间进行SPI通信时要使用的MSSP模块（见图2-4）。

所选MSSP模块必须配置为模式0（SPI主模式）。

图2-4： LoRaWAN™ SPI模块选择



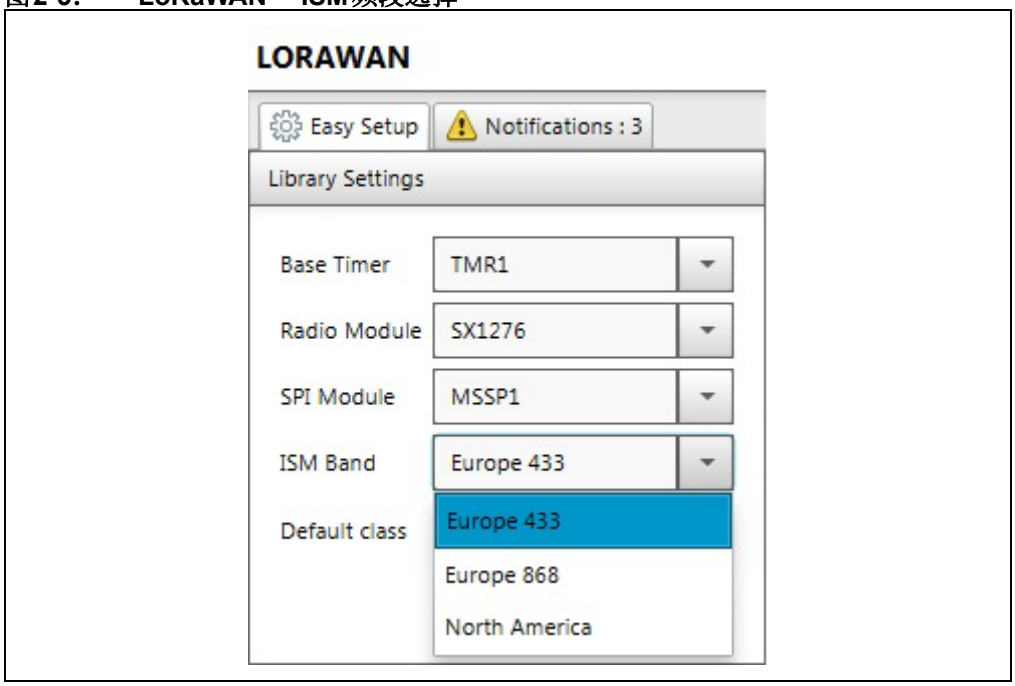
2.1.1.4 ISM频段选择

LoRaWAN协议栈目前支持以下两个地区：欧洲和北美，具体规范如下：

- 欧洲采用433和868 MHz频段
- 北美采用915 MHz频段

用户必须选择所需ISM频段（图2-5）。

图2-5： LoRaWAN™ ISM频段选择



2.1.1.5 默认LoRaWAN类别选择

Microchip LoRaWAN协议栈支持A类和C类。用户可从Default class下拉菜单内的两个类别中选择哪个类别应在启动时起作用。此外，还可以使用类别更改API在运行时更改类别：

- A类——器件在启动时应初始化为A类，并且应作为A类器件工作
- C类——器件在启动时应初始化为C类，并且应作为C类器件工作

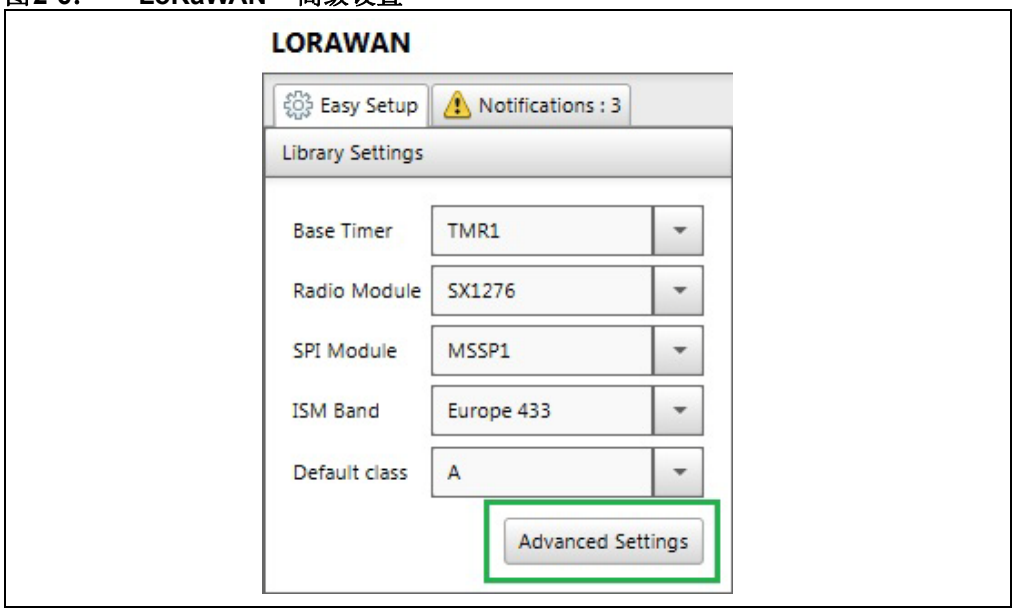
可以使用LoRaWAN协议栈提供的API在运行时更改工作类别（更多详细信息，请参见**第4章“LoRaWAN™协议栈API”**）。

© 2017 Microchip Technology Inc.

2.1.2 LoRaWAN 高级设置

单击**Advanced Settings**按钮可访问LoRaWAN协议栈的高级设置，如图2-6所示。此操作将为用户提供一个向导菜单，从而可针对EU 433/868添加或删除通道、针对NA 915禁止和使能通道以及对LoRaWAN协议栈进行各种调整。

图2-6： LoRaWAN™ 高级设置



2.1.2.1 欧洲433 MHz高级设置

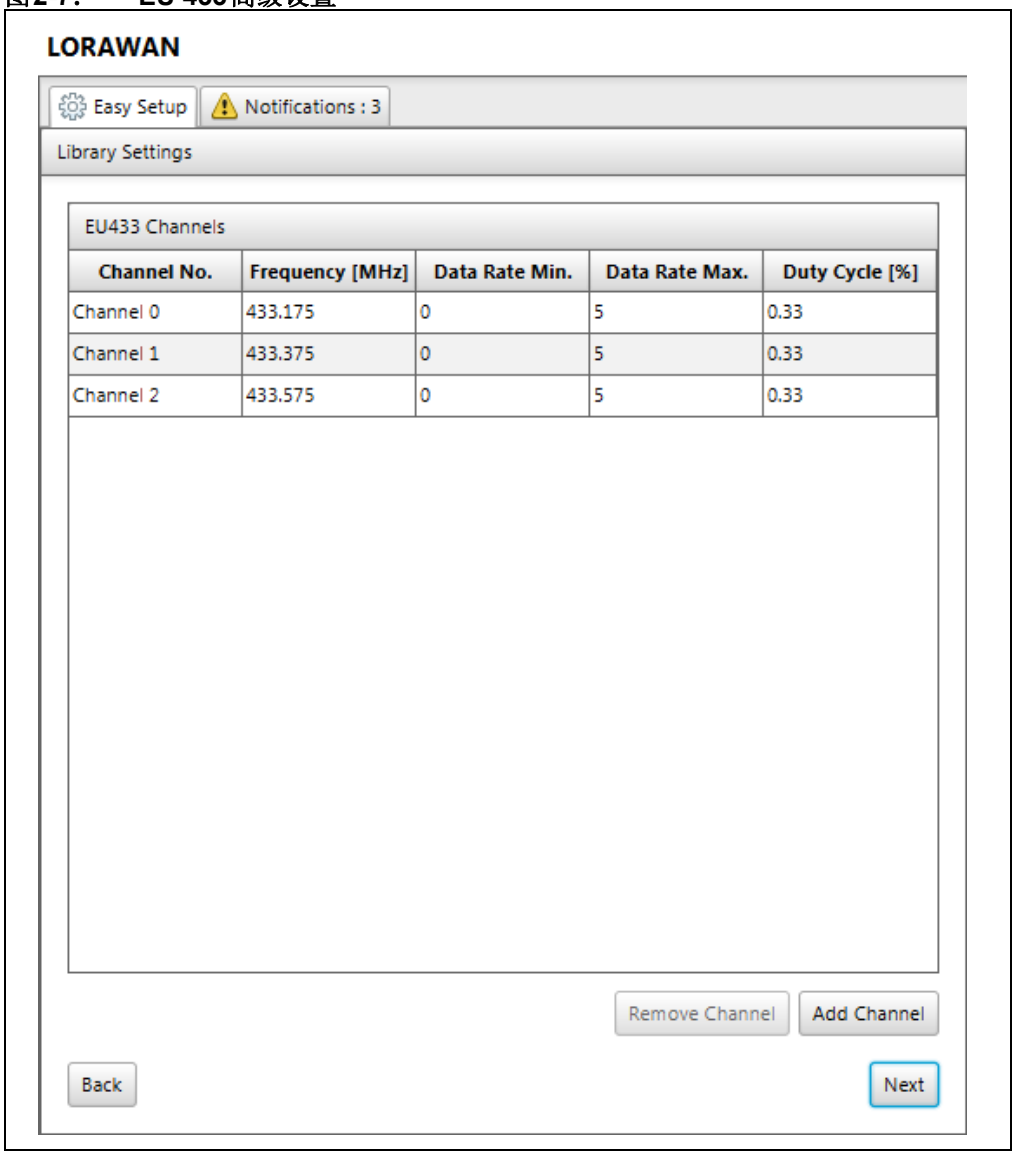
单击EU 433 ISM频段对应的**Advanced Settings**按钮后，将显示**Library Settings**（库设置）选项卡。Advanced Settings的向导菜单的第一个选项卡显示EU433 Channels（EU433通道）表（图2-7）。表中包括以下各行：

- Channel Number（通道编号）
- Frequency（频率）
- Data Rate Minimum（最小数据速率）
- Data Rate Maximum（最大数据速率）
- Duty Cycle（占空比）

上述各行表示每个通道的可配置参数。表中已包括前三个默认通道。LoRaWAN规范V1.0文档中规定了这些默认通道及其设置。默认通道的频率固定，不可更改。最小和最大数据速率范围可以调整，可接受的数值范围为0至7（最大数据速率必须大于最小数据速率）。

每次创建或删除新通道时都需要调整占空比，以便遵循欧洲电信标准协会（European Telecommunications Standards Institute，ETSI）发布的严格限制。

图2-7： EU 433高级设置

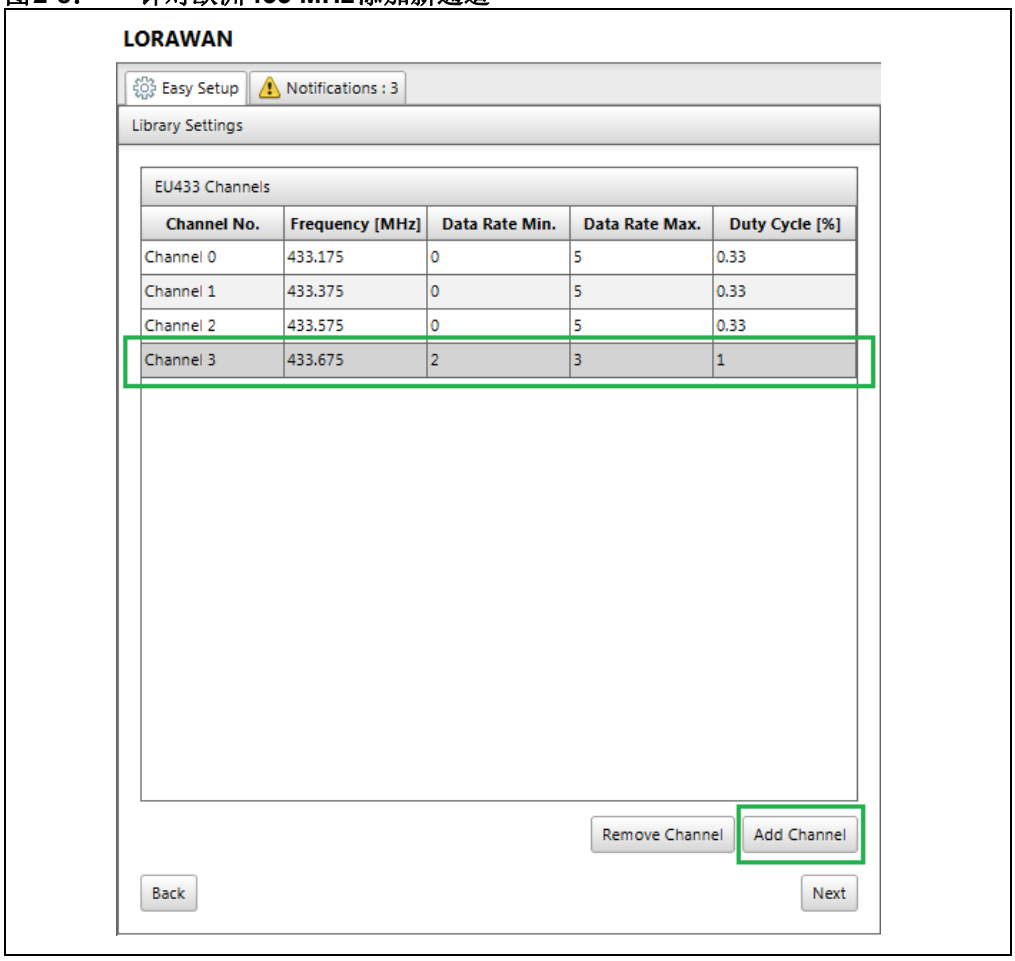


© 2017 Microchip Technology Inc.

单击**Add Channel**（添加通道）按钮可向通道表中添加新通道（图2-8）。最多可添加13个通道（根据LoRaWAN规范V1.0，可接受的最大数量为16个通道，其中包括默认通道）。添加到表中的新条目包含以下可配置参数：

- **频率**——默认情况下，初始值为433.175 MHz。可接受的EU 433 ISM频段范围为433.175 MHz至434.665 MHz。如果用户插入的值不在该范围内，单元格将拒绝该值并指示值无效，同时会以红色背景闪烁。
- **数据速率**——表示通信时使用的无线电参数。该数字介于0和7之间，对应于250 bps和50000 bps之间的物理比特率（更多详细信息，请参见LoRaWAN规范V1.0文档）。可提供最小值和最大值；如果用户插入的值不在该范围内（或者最小值大于最大值），则单元格将拒绝该值并指示值无效，同时会以红色背景闪烁。
- **占空比**——表示发射时间。例如：对于100秒的周期而言，1%的占空比表示终端设备有1秒处于发射状态，其余99秒保持待机状态。可接受的占空比范围为0至100%。如果用户插入的值不在该范围内，单元格将拒绝该值并指示值无效，同时会以红色背景闪烁。

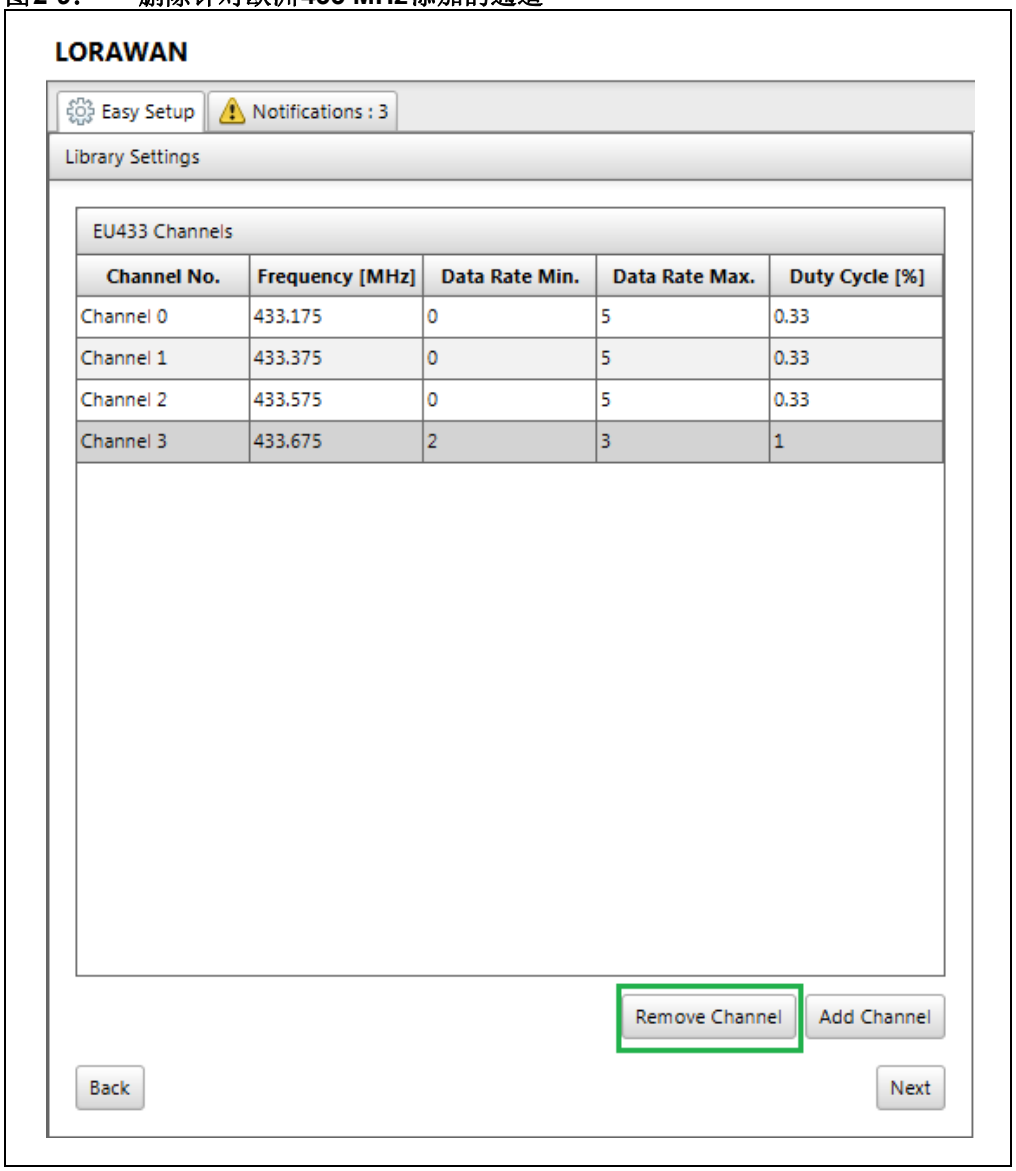
图2-8：　针对欧洲433 MHz添加新通道



有关通道和数据速率的更多详细信息，请参见LoRaWAN规范V1.0文档。

可从表中删除之前插入的任何通道，方法是选中相应通道并单击**Remove Channel**（删除通道）按钮（图2-9）。前三个默认通道不可删除。

图2-9： 删除针对欧洲433 MHz添加的通道

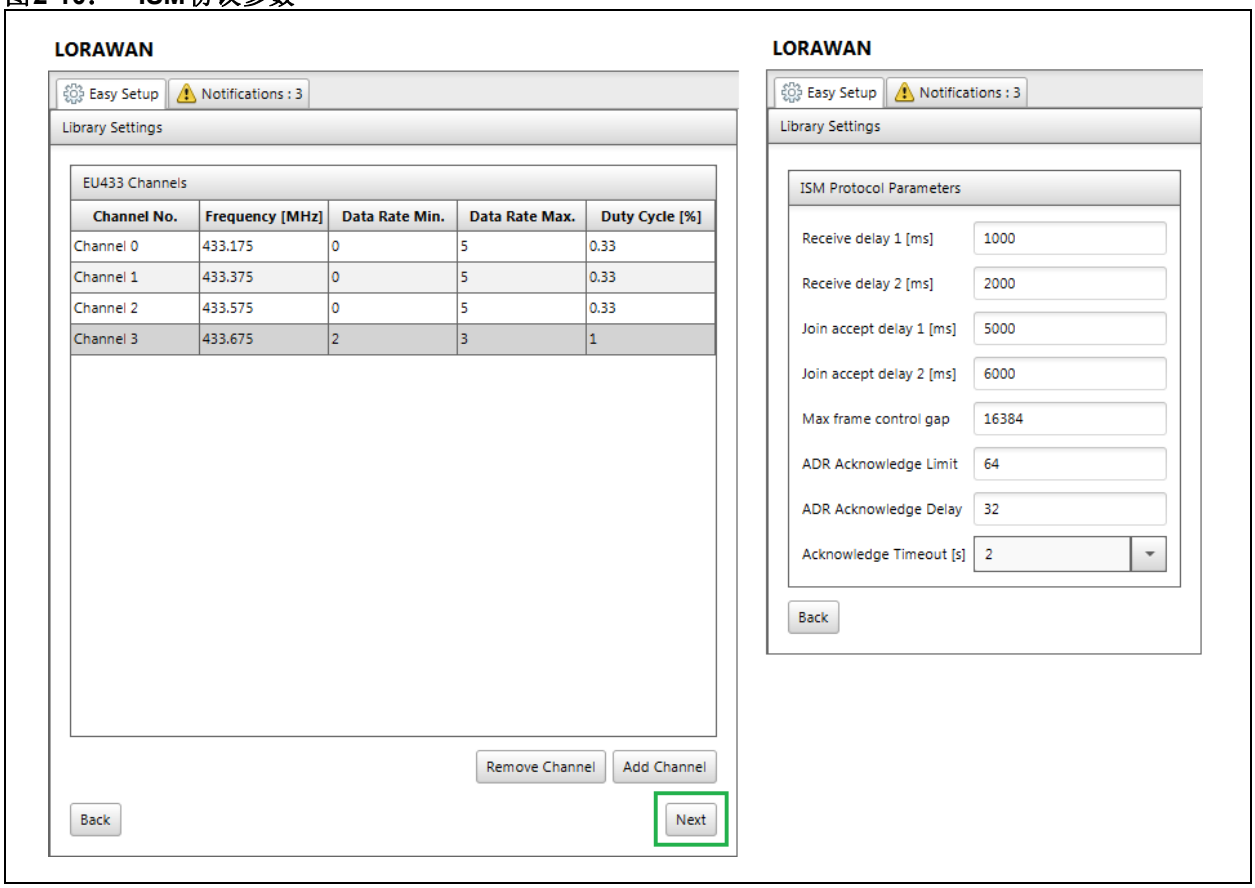


有关通道和数据速率的更多详细信息，请参见LoRaWAN规范V1.0文档。

© 2017 Microchip Technology Inc.

单击**Next**（下一步）按钮可访问EU433 Advanced Settings的第二个选项卡。该选项卡允许用户配置LoRaWAN协议栈参数，例如，接收延时、加入接受延时或确认超时。这些参数已载入默认值。

图2-10： ISM协议参数



> **注：** 仅限熟练掌握LoRa®技术的用户更改这些值。如果为这些参数设置的值不正确，则可能导致LoRaWAN™ 协议栈无法正常工作。

ISM协议参数设置完成后，用户可按下**Back**（后退）按钮返回到上一个选项卡——**EU433 Channels**。用户再次按下**Back**按钮后，将返回到LoRaWAN库的Easy Setup（简单设置）视图。

2.1.2.2 欧洲868 MHz高级设置

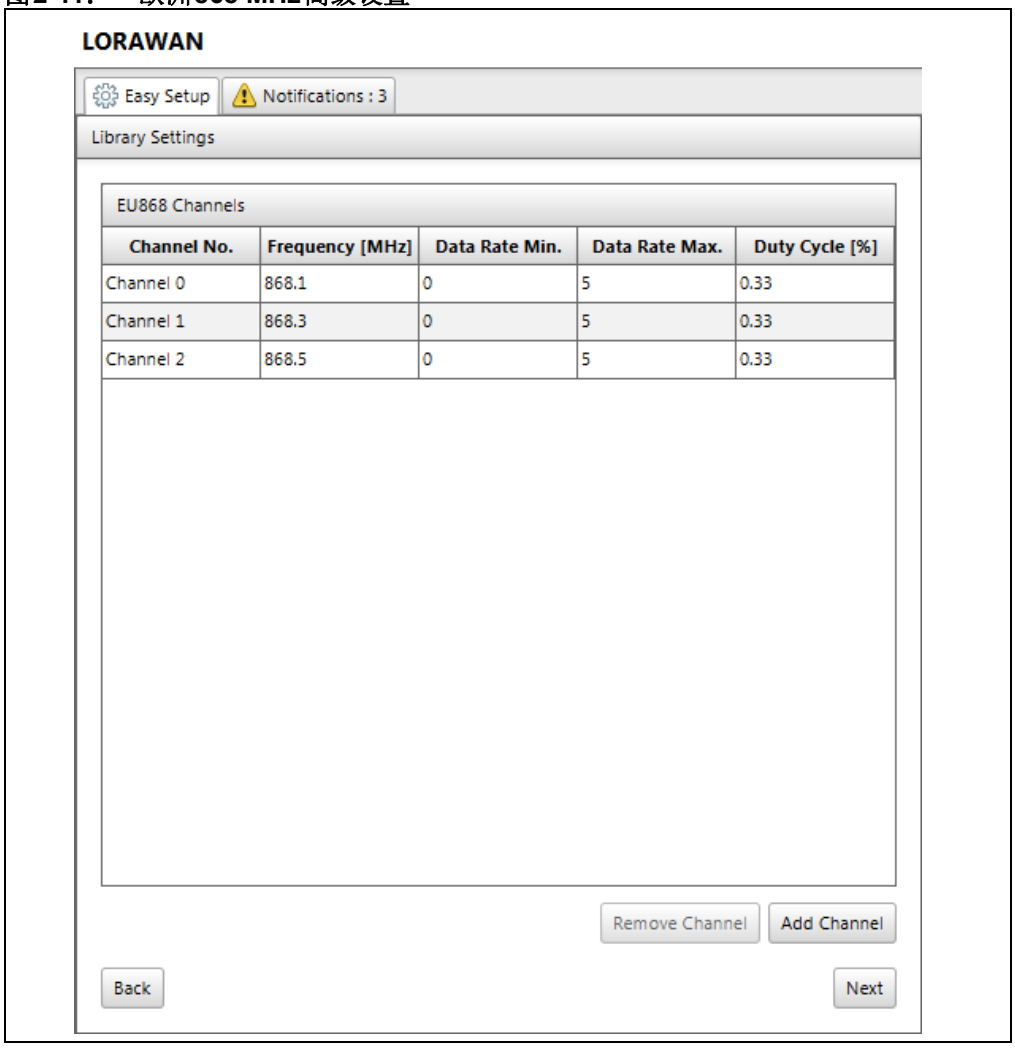
单击EU868 ISM频段对应的**Advanced Settings**按钮后，将显示**Library Settings**选项卡。Advanced Settings的向导菜单的第一个选项卡显示EU868 channels（EU868通道）表，如图2-11所示。表中包括以下各行：

- 通道编号
- 频率
- 最小数据速率
- 最大数据速率
- 占空比

上述各行表示每个通道的可配置参数。表中已包括前三个默认通道。LoRaWAN规范V1.0文档中规定了这些默认通道及其设置。默认通道的频率固定，不可更改。最小和最大数据速率范围可以调整，可接受的数值范围为0至7（最大数据速率必须大于最小数据速率）。

每次创建或删除新通道时都需要调整占空比，以便遵循欧洲电信标准协会（ETSI）发布的严格限制。

图2-11： 欧洲868 MHz高级设置

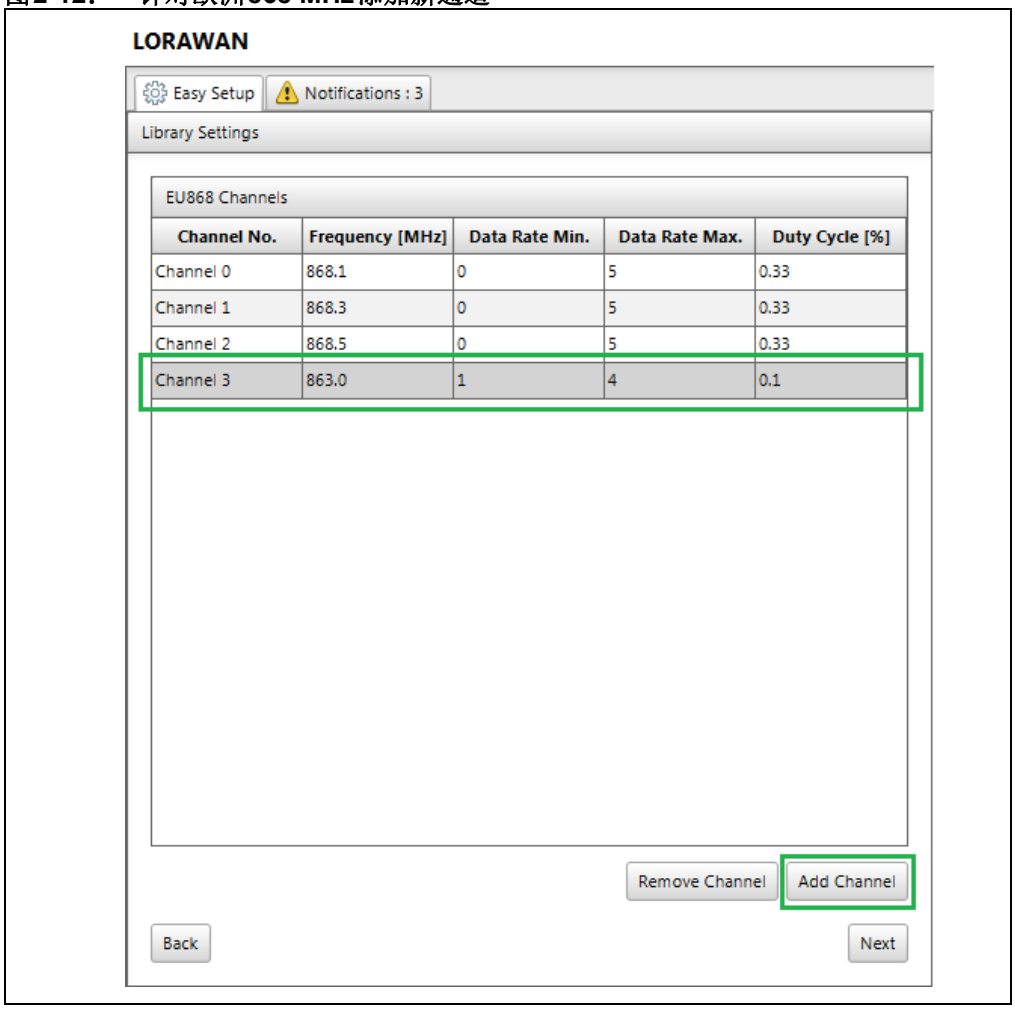


© 2017 Microchip Technology Inc.

单击**Add Channel**按钮可向通道表中添加新通道（图2-12）。最多可添加13个通道（根据LoRaWAN规范V1.0，可接受的最大数量为16个通道，其中包括默认通道）。添加到表中的新条目包含以下可配置参数：

- **频率**——默认情况下，初始值为863.0 MHz。可接受的EU 868 ISM频段范围为863.0 MHz至870.0 MHz。如果用户插入的值不在该范围内，单元格将拒绝该值并指示值无效，同时会以红色背景闪烁。
- **数据速率**——表示通信时使用的无线电参数。该数字介于0和7之间，对应于250 bps和50000 bps之间的物理比特率（更多详细信息，请参见LoRaWAN规范V1.0文档）。可提供最小值和最大值；如果用户插入的值不在该范围内（或者最小值大于最大值），则单元格将拒绝该值并指示值无效，同时会以红色背景闪烁。
- **占空比**——表示发射时间。例如：对于100秒的周期而言，1%的占空比表示终端设备有1秒处于发射状态，其余99秒保持待机状态。可接受的占空比范围为0至100%。如果用户插入的值不在该范围内，单元格将拒绝该值并指示值无效，同时会以红色背景闪烁。

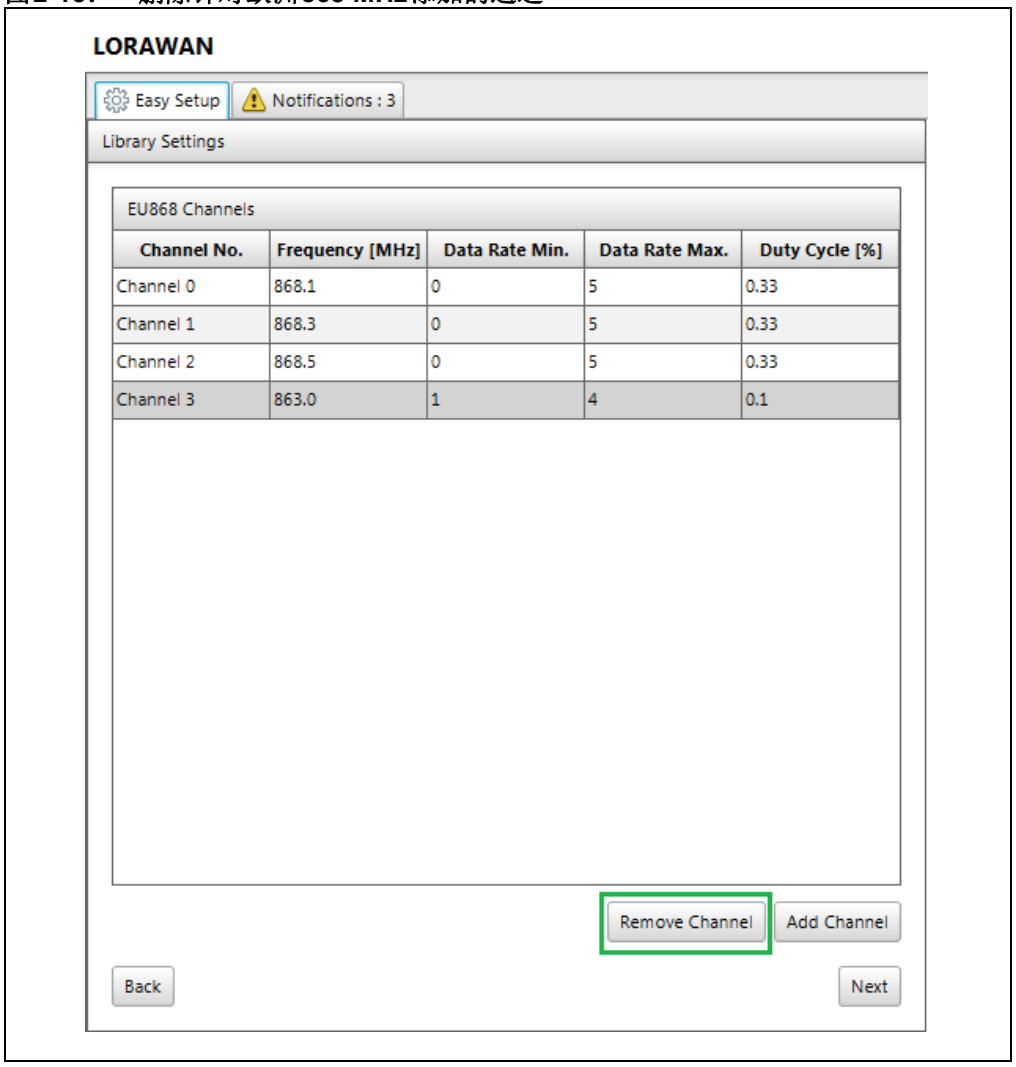
图2-12：　针对欧洲868 MHz添加新通道



有关通道和数据速率的更多详细信息，请参见LoRaWAN规范V1.0文档。

可从表中删除之前插入的任何通道，方法是选中相应通道并单击**Remove Channel**按钮，如图2-13所示。前三个默认通道不可删除。

图2-13： 删除针对欧洲868 MHz添加的通道

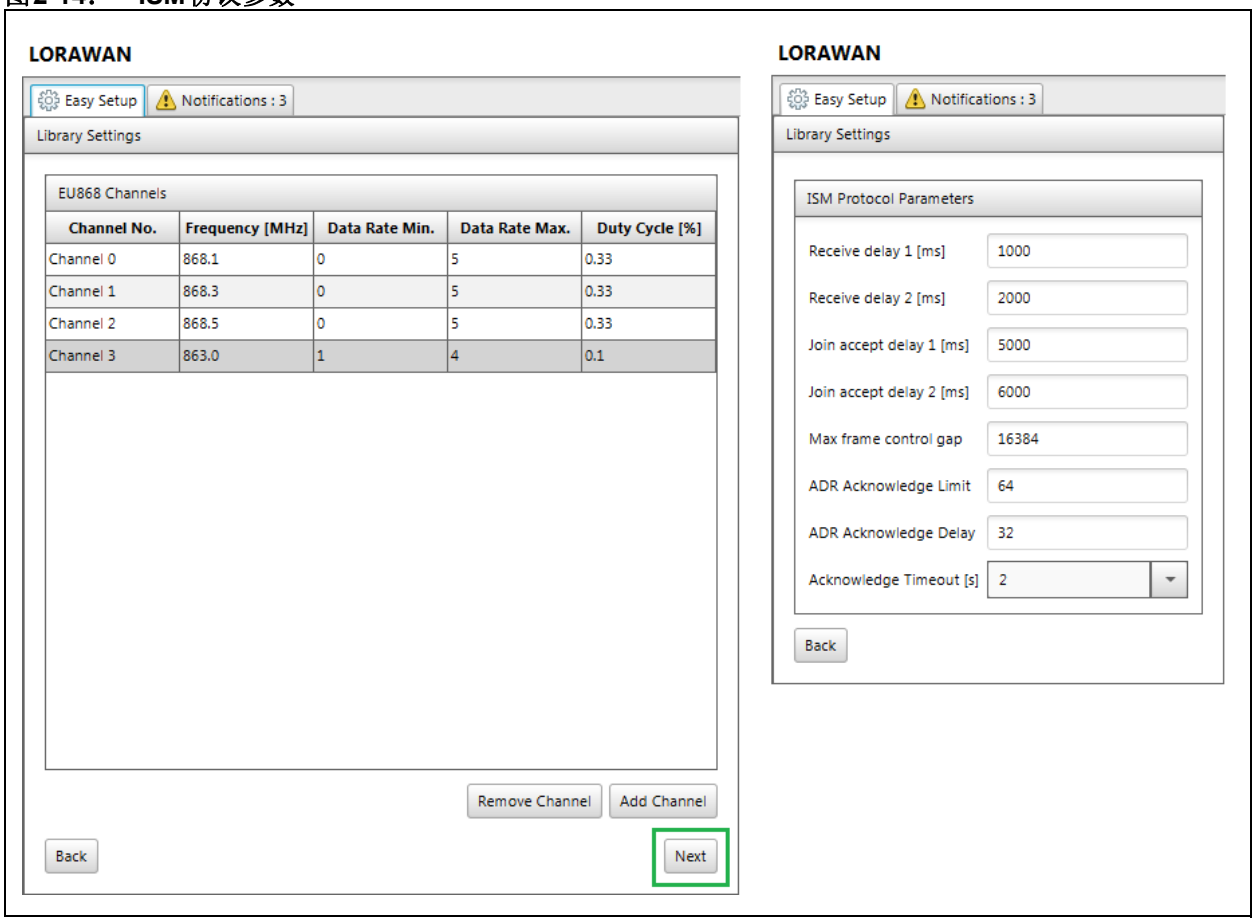


有关通道和数据速率的更多详细信息，请参见LoRaWAN规范V1.0文档。

© 2017 Microchip Technology Inc.

单击**Next**按钮可访问EU868 Advanced Settings的第二个选项卡。该选项卡允许用户配置LoRaWAN协议栈参数，例如，接收延时、加入接受延时或确认超时。这些参数已载入默认值。

图2-14： ISM协议参数



> **注：** 仅限熟练掌握LoRa®技术的用户更改这些值。如果为这些参数设置的值不正确，则可能导致LoRaWAN™ 协议栈无法正常工作。

ISM协议参数设置完成后，用户可按下**Back**按钮返回到上一个选项卡——**EU868 Channels**。用户再次按下**Back**按钮后，将返回到LoRaWAN库的Easy Setup视图。

2.1.2.3 北美915 MHz高级设置

对于北美915 MHz，共有72个通道，这些通道分为以下两类：0至63和64至71。LoRaWAN规范V1.0文档中规定了所有通道的频率。默认情况下，72个通道均已使能。

清除每个通道Active（活动）列上的复选框可禁止通道，选中该**Active**列上的复选框可再次使能通道。

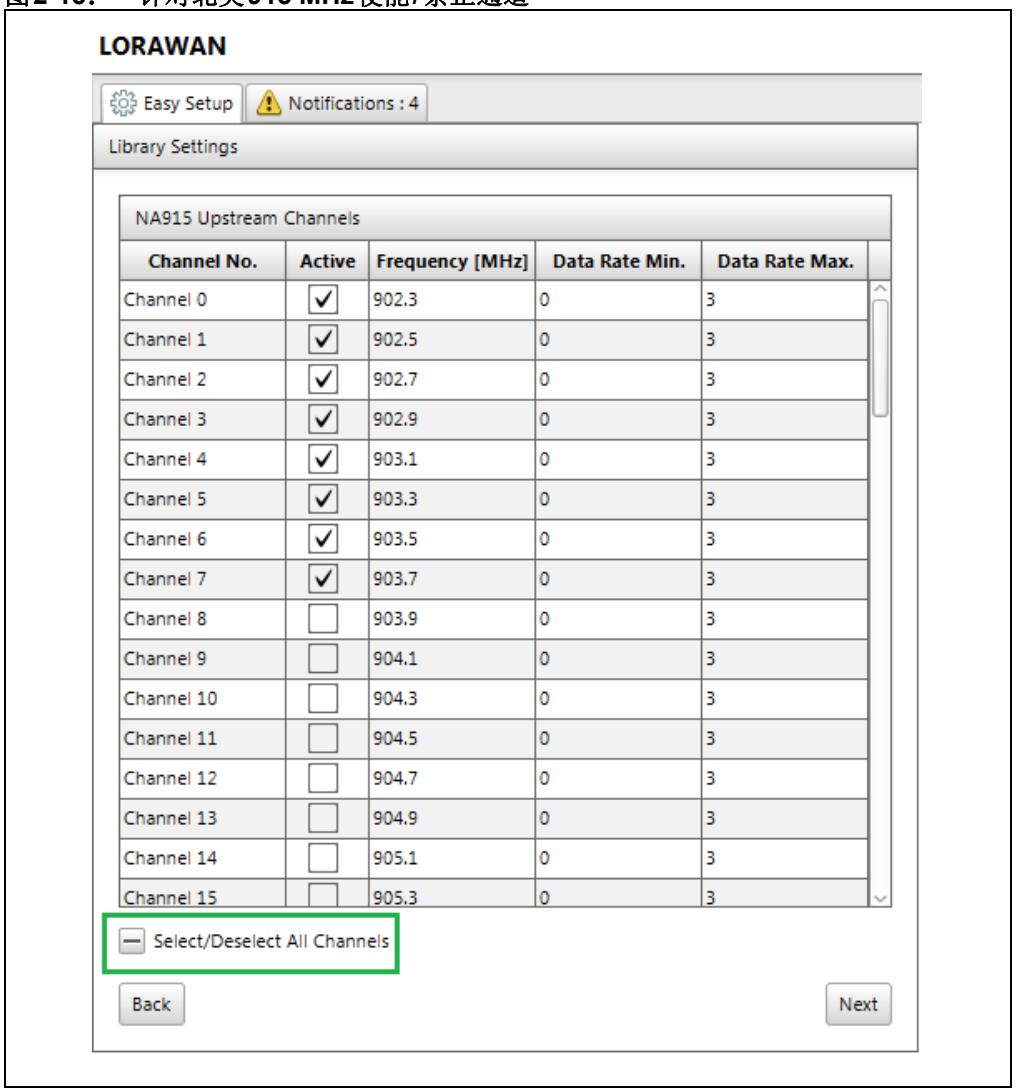
对于通道0–63，可更改Data Rate Min.（最小数据速率）和Data Rate Max.（最大数据速率）的值。最小数据速率和最大数据速率的范围为0至3，对应于980 bps至5470 bps。

表中有一个Select/Deselect All Channels（选中/取消选中所有通道）按钮，可用于针对北美使能或禁止所有通道（见图2-15）。

对于通道64–71，数据速率固定为4（对应于12500 bps），因此，Data Rate Min.的值等于Data Rate Max.的值，这些单元格不可编辑。

有关通道和数据速率的更多详细信息，请参见LoRaWAN规范V1.0文档。

图2-15： 针对北美915 MHz使能/禁止通道

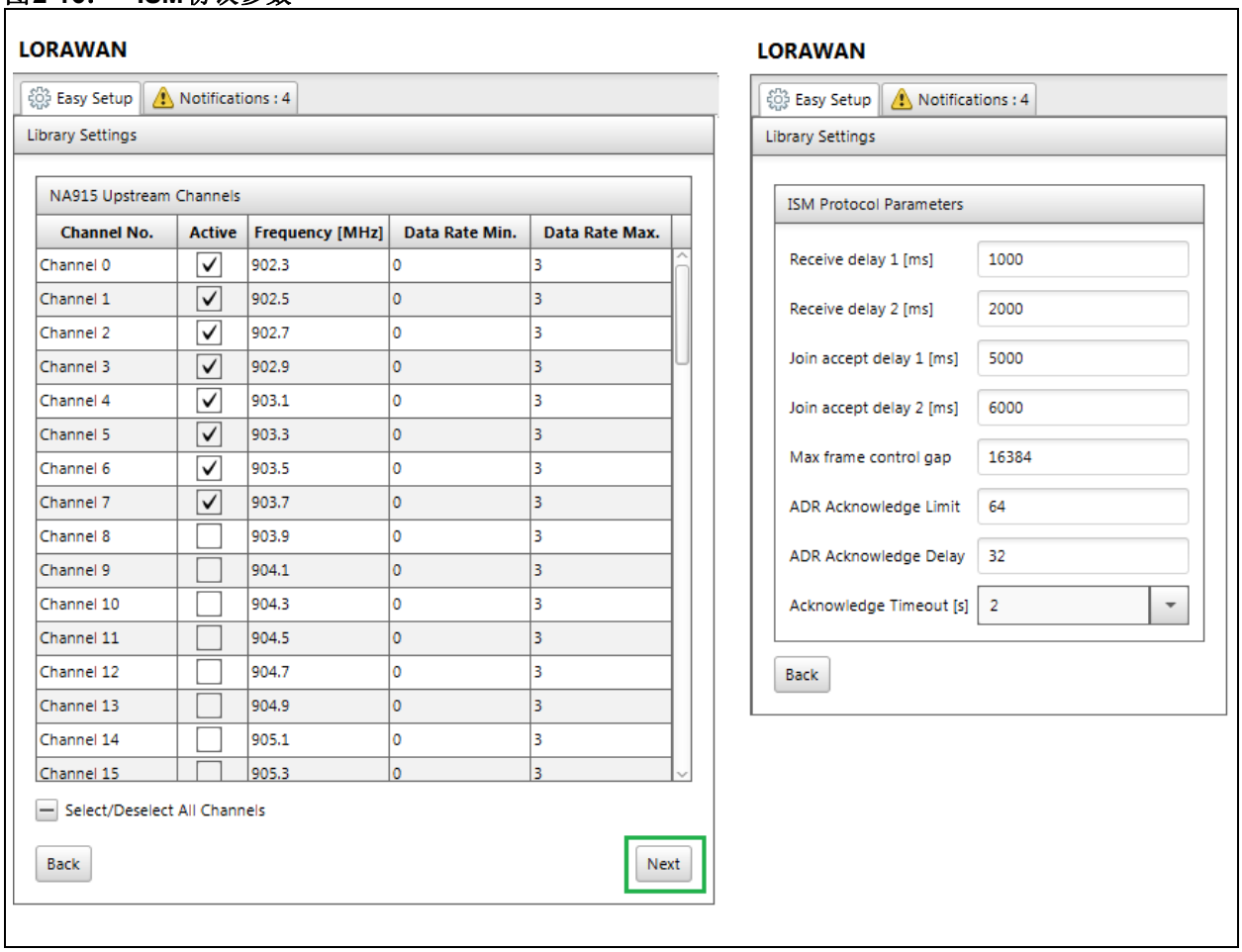


Channel No.	Active	Frequency [MHz]	Data Rate Min.	Data Rate Max.
Channel 0	✓	902.3	0	3
Channel 1	✓	902.5	0	3
Channel 2	✓	902.7	0	3
Channel 3	✓	902.9	0	3
Channel 4	✓	903.1	0	3
Channel 5	✓	903.3	0	3
Channel 6	✓	903.5	0	3
Channel 7	✓	903.7	0	3
Channel 8		903.9	0	3
Channel 9		904.1	0	3
Channel 10		904.3	0	3
Channel 11		904.5	0	3
Channel 12		904.7	0	3
Channel 13		904.9	0	3
Channel 14		905.1	0	3
Channel 15		905.3	0	3

© 2017 Microchip Technology Inc.

单击**Next**按钮可访问北美915 MHz Advanced Settings的第二个选项卡。该选项卡允许用户配置LoRaWAN协议栈参数，例如，接收延时、加入接受延时或确认超时。这些参数已载入默认值。

图2-16： ISM协议参数



> **注：** 仅限熟练掌握LoRa®技术的用户更改这些值。如果为这些参数设置的值不正确，则可能导致LoRaWAN™ 协议栈无法正常工作。

ISM协议参数设置完成后，用户可按下**Back**按钮返回到上一个选项卡——**NA915 Channels**（NA915通道）。用户再次按下**Back**按钮后，将返回到LoRaWAN库的Easy Setup视图。

注：

© 2017 Microchip Technology Inc.

第3章 运行在RN2XX3模块上的LoRaWAN™库插件

3.1 概述

Microchip RN2483和RN2903模块上已存在LoRaWAN协议栈。本章介绍了为生成LoRaWAN协议栈以及在Microchip RN2XX3模块上运行而创建的项目配置。

可以在RN2483和RN2903上进行相同的配置；但在LoRaWAN库基本配置期间选择的ISM频段有所不同。

RN2483和RN2903模块的核心器件包括PIC18LF46K22器件和Semtech SX1276无线电收发器。因此，需为该特定系统（PIC18LF46K22和SX1276）进行项目配置。

下文中将使用RN2903（用于北美）作为示例。

> **注：** RN2XX3模块作为标准模块、法规认证模块和LoRa联盟认证模块销售。这些认证参考了标准生产固件，因此使用自定义MCC固件覆盖后将使这些认证无效。

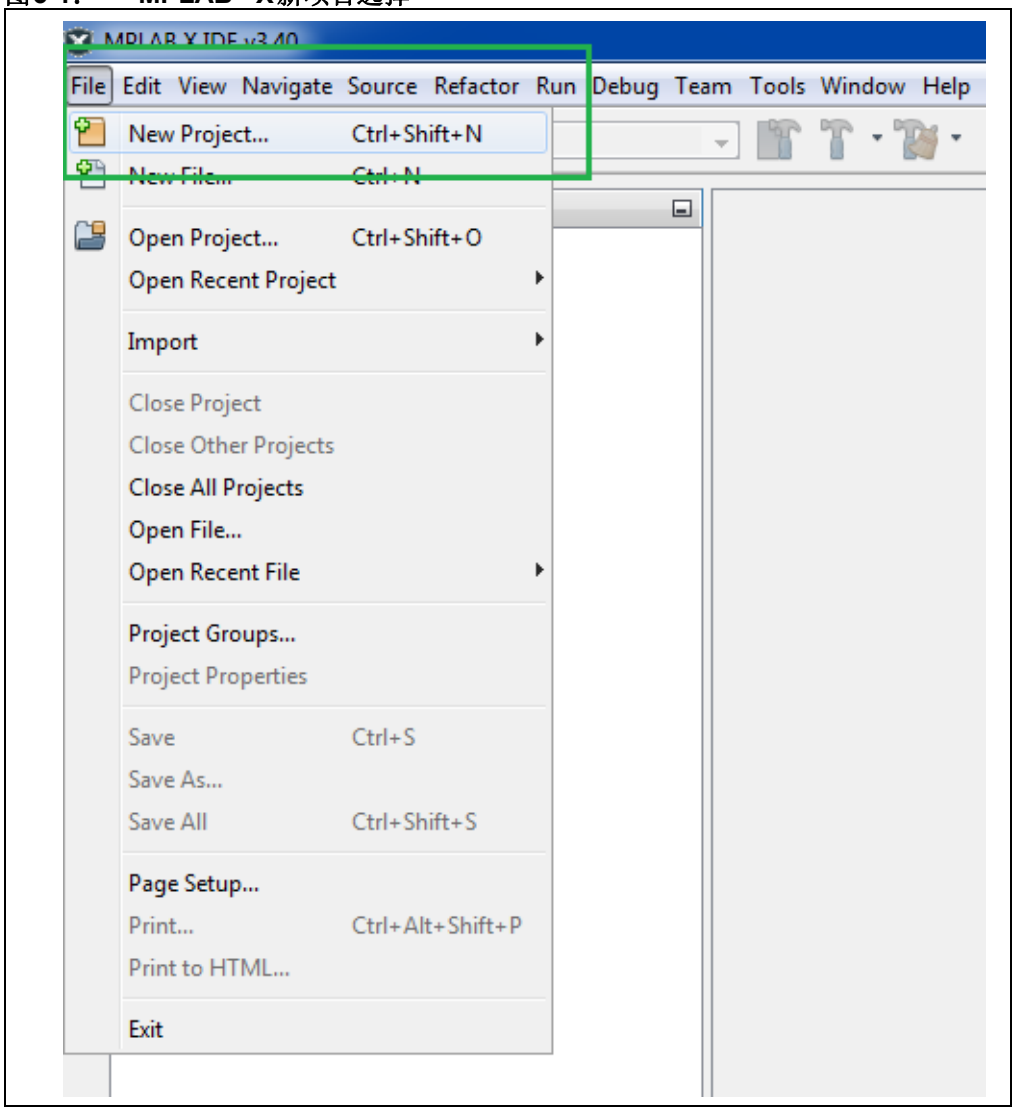
3.2 RN2XX3模块上的LoRaWAN配置

3.2.1 LoRaWAN项目创建

必须创建并设置新的MPLAB® X项目。具体步骤如下：

1. 在MPLAB® X中，单击*File > New Project*（文件 > 新建项目）（见图3-1）。

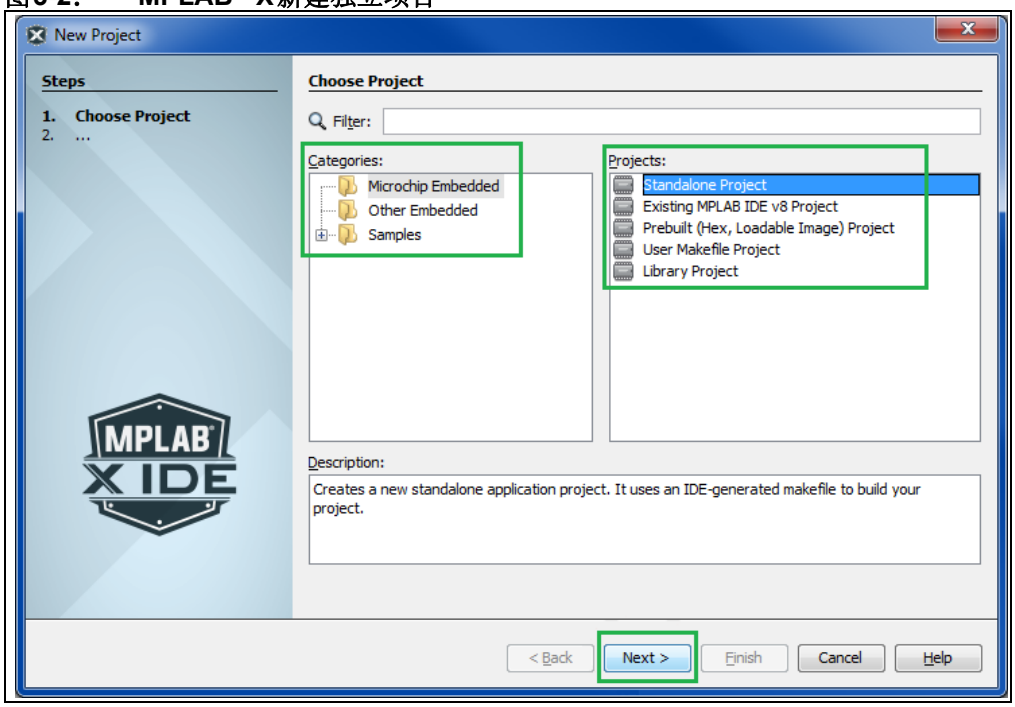
图3-1： MPLAB® X新项目选择



© 2017 Microchip Technology Inc.

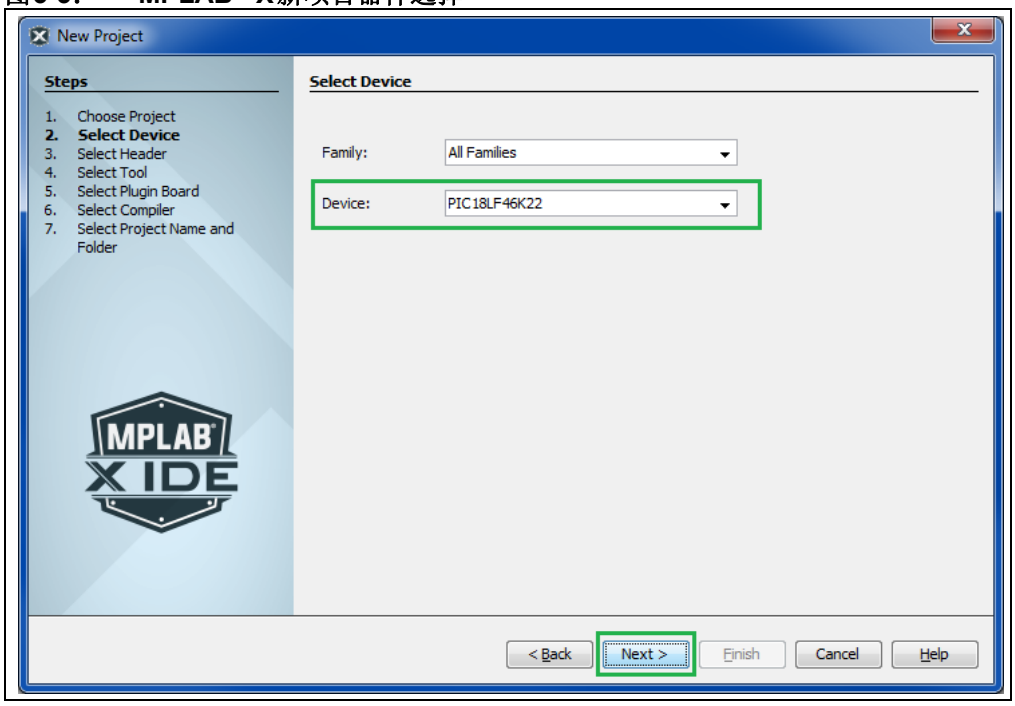
2. 在Categories（类别）窗口下选择Microchip Embedded（Microchip已安装工具），在Projects（项目）窗口下选择Standalone Project（独立项目），然后单击**Next**（见图3-2）。

图3-2： MPLAB® X新建独立项目



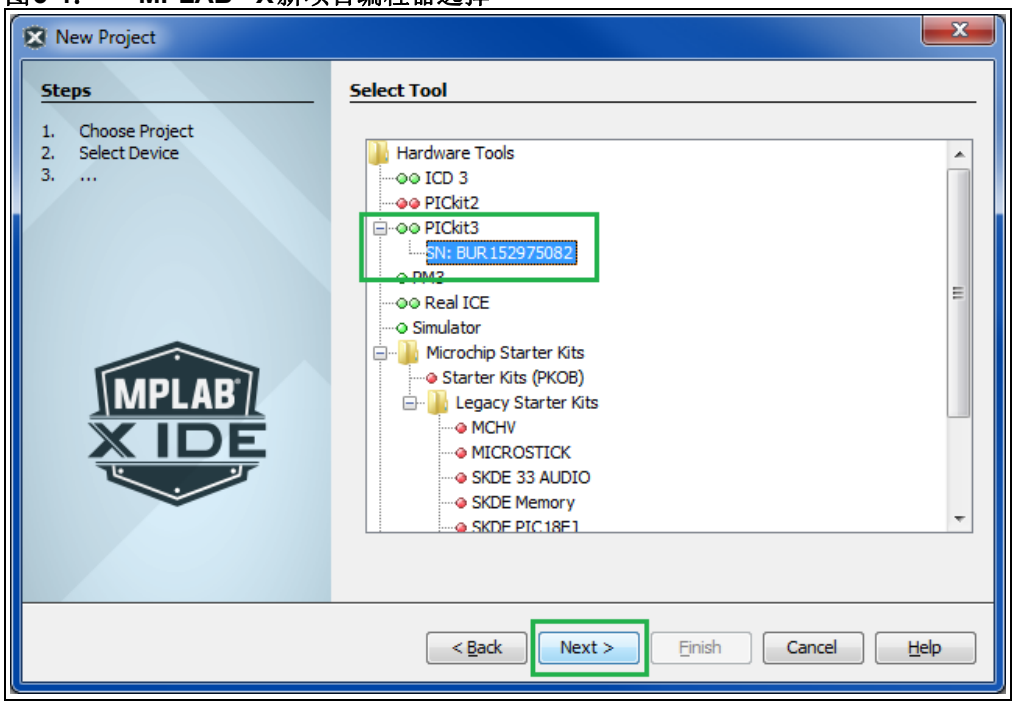
3. 从Device（器件）下拉列表中选择PIC18LF46K22，然后按下**Next**按钮，如图3-3所示。

图3-3： MPLAB® X新项目器件选择



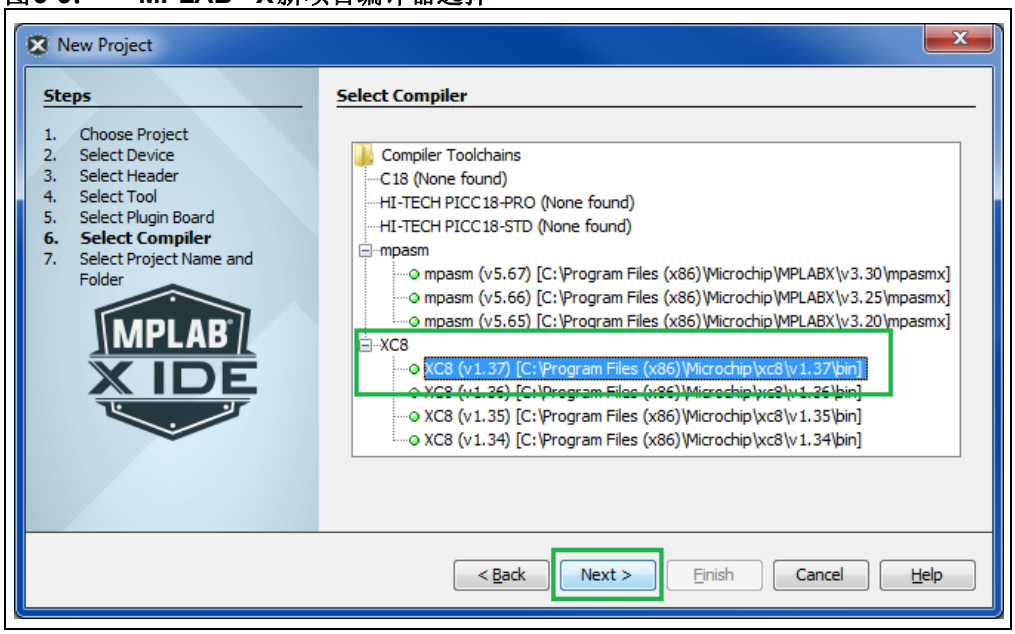
4. 从Hardware Tools（硬件工具）树中选择所使用的编程器，然后单击**Next**，如图3-4中所示。如果没有编程器或调试工具物理连接到PC，则可以使用软件模拟器来代替。

图3-4： MPLAB® X新项目编程器选择



5. 选择要使用的编译器。LoRaWAN库需要XC8 v1.37（或更高版本）。选择XC8（v1.37），然后单击**Next**，如图3-5所示。

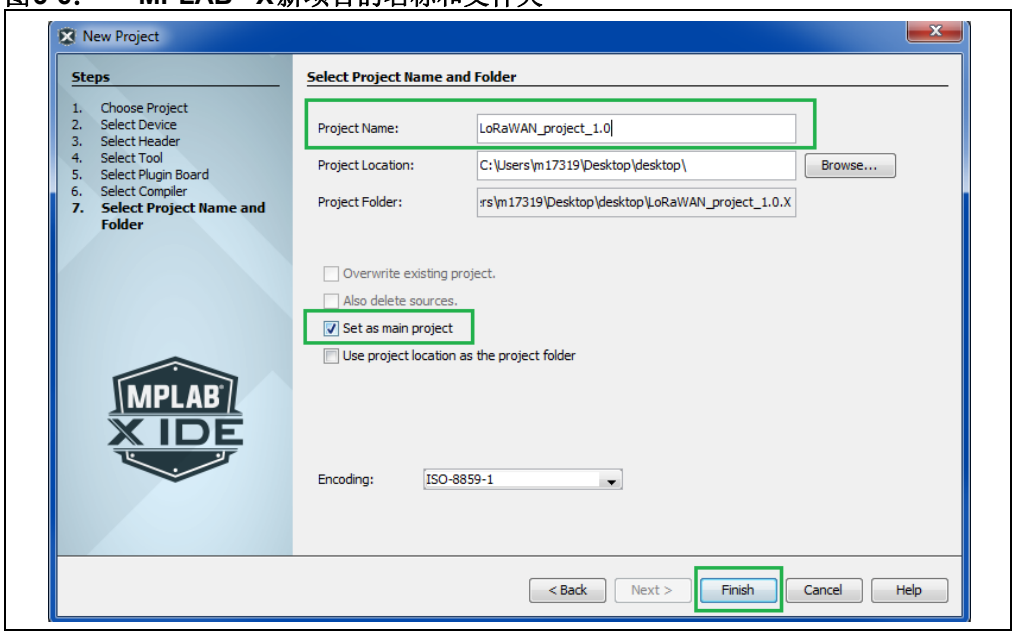
图3-5： MPLAB® X新项目编译器选择



© 2017 Microchip Technology Inc.

6. 设置项目的名称，并确保选中“Set as main project”（设置为主项目）复选框。单击**Finish**（完成）按钮；这将结束新项目的创建过程。

图3-6： MPLAB® X新项目的名称和文件夹

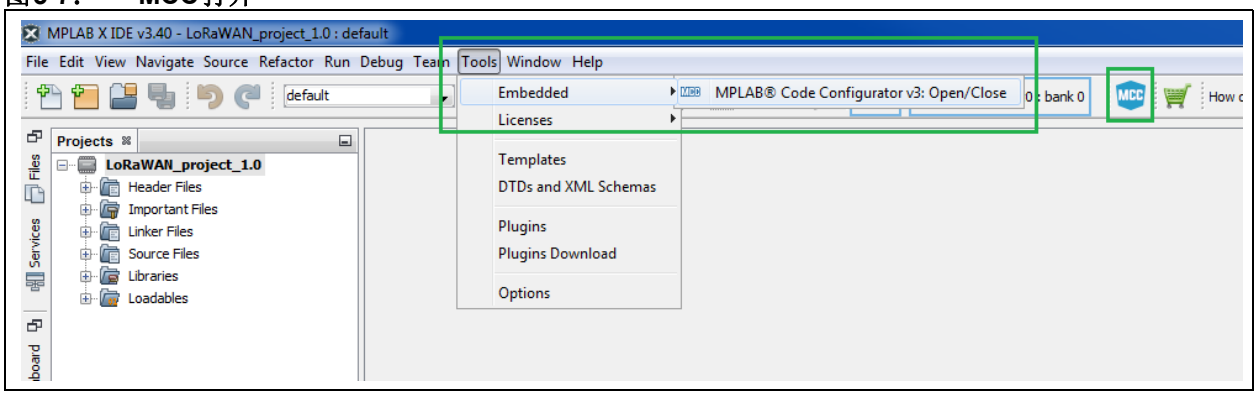


3.2.2 LoRaWAN基本配置

创建LoRaWAN基本配置的步骤如下。

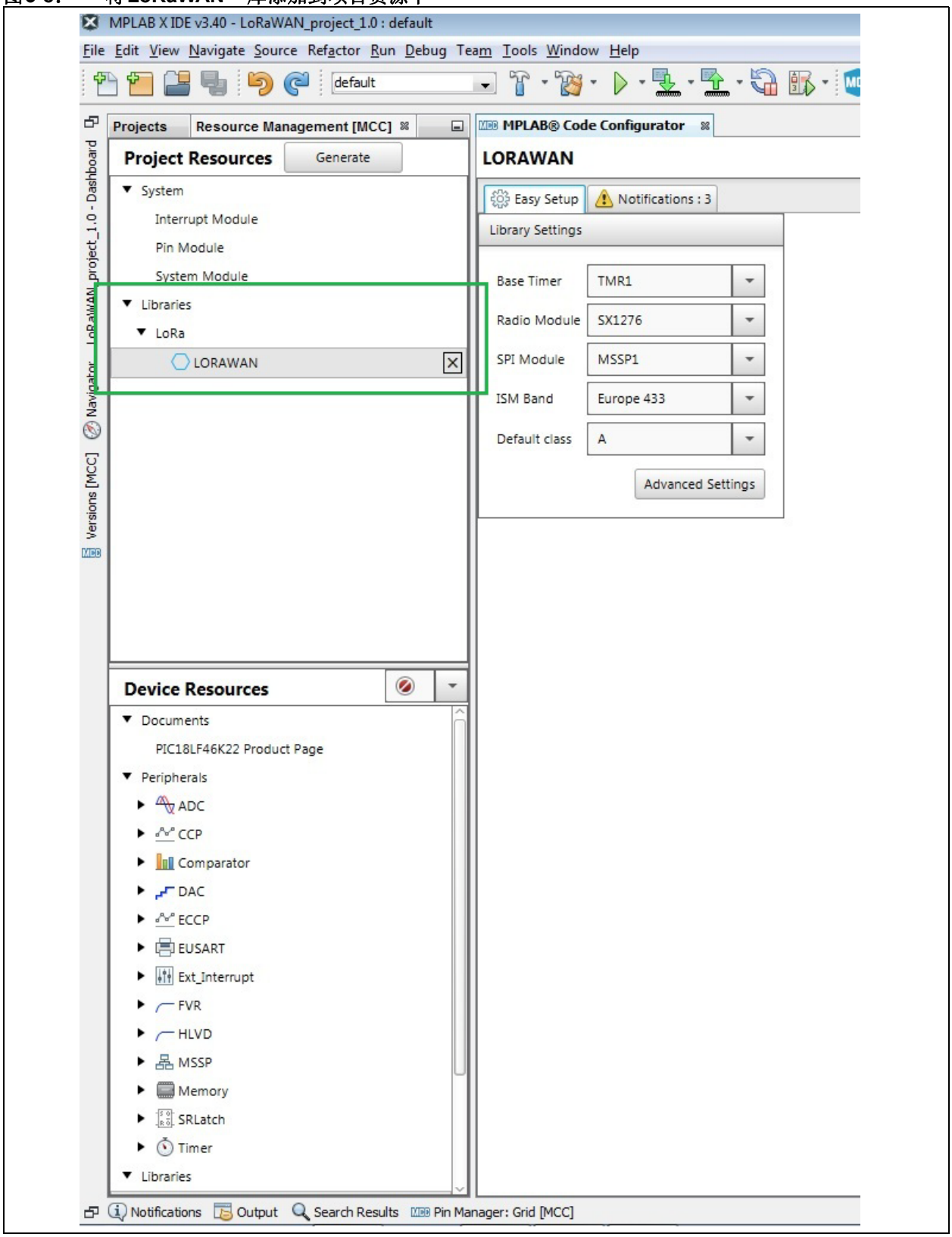
1. 打开MPLAB®代码配置器（MPLAB® Code Configurator，MCC），以创建项目的配置。在MPLAB X中，单击*Tools > Embedded > MPLAB® Code Configurator v3: Open/Close*（工具 > 已安装工具 > MPLAB®代码配置器v3：打开/关闭）或者单击MPLAB X工具栏上的MCC图标 （见图3-7）。

图3-7： MCC打开



2. 添加LoRaWAN库。从Device Resources（器件资源）窗口中，展开*Libraries > LoRa*（库 > LoRa），然后双击**LoRaWAN**。该库将添加到Project Resources（项目资源）中（见图3-8）。

图3-8：　将LoRaWAN™库添加到项目资源中



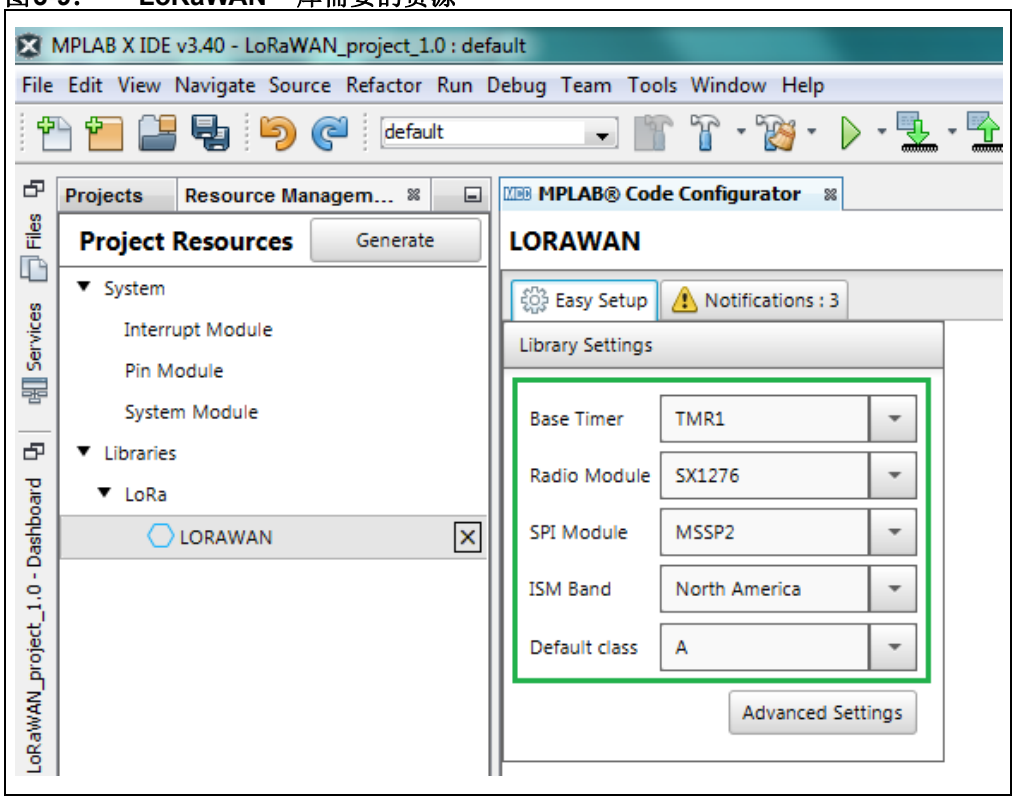
© 2017 Microchip Technology Inc.

3. 选择LoRaWAN协议栈所需的资源（基本定时器，无线电模块，SPI和ISM频段）。对LoRaWAN库进行以下设置，如图3-9所示：

- Base Timer：TMR1
- Radio Module：SX1276
- SPI Module：MSSP2
- ISM Band：North America（北美）
- Default Class：A

> **注：** 应根据使用LoRaWAN协议栈的终端设备所在的地区来选择ISM频段。如果用户想在欧洲境内运行基于LoRaWAN的应用，则选择的ISM频段应为欧洲433 MHz（列表中的EU433）或欧洲868 MHz（列表中的EU868）。

图3-9： LoRaWAN™库需要的资源

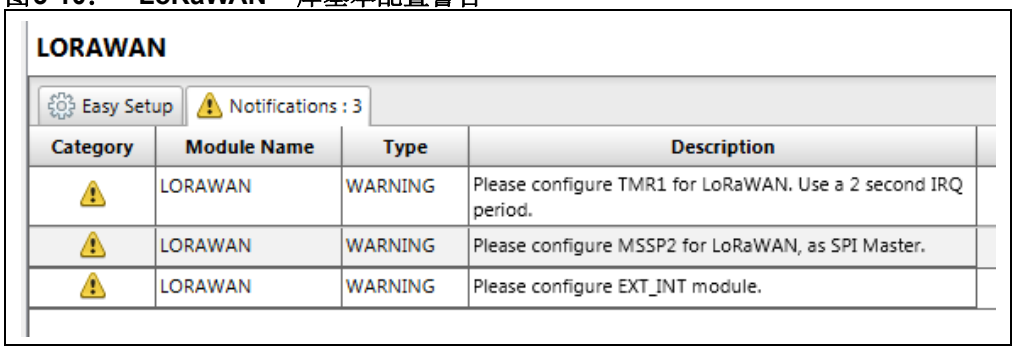


完成上述设置后，**Notifications**（通知）选项卡上即会显示三条通知。单击该选项卡可查看这些通知（如图3-10所示）。它们将向用户发送如下相关警告：

- 针对LoRaWAN将TMR1配置为使用2秒中断请求周期
- 针对LoRaWAN将MSSP2配置为SPI主模式
- 配置EXT_INT（外部中断）模块

所有这些设置将在**第3.2.3节“LoRaWAN RN2XX3配置”**中进一步解释。

图3-10： LoRaWAN™库基本配置警告



3.2.3 LoRaWAN RN2XX3配置

3.2.3.1 概述

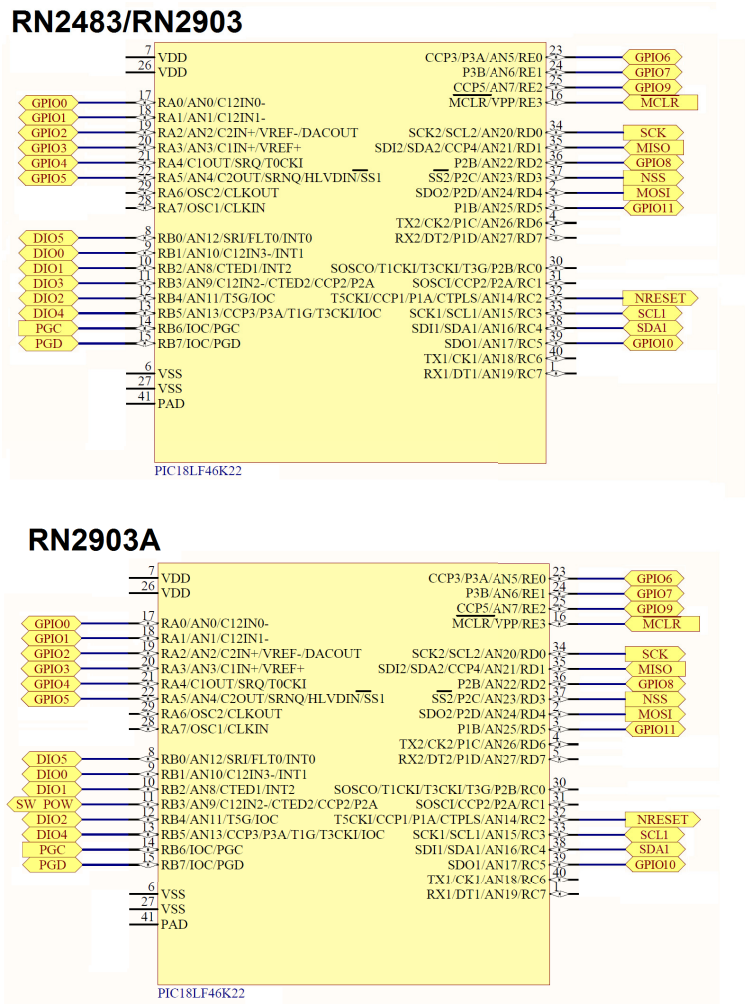
本章中介绍的配置与RN2XX3模块（RN2483和RN2903）上的硬件和物理硬件连接密切相关。

RN2903上的硬件连接如下所示（图3-11）。关键部分包括：

- SPI引脚（MSSP引脚）
- NSS（片选）
- NRESET引脚
- DIOx引脚
- SW_POW引脚（仅适用于RN2903A）

© 2017 Microchip Technology Inc.

图3-11： RN2903/RN2483硬件连接



> **注：** SCL1和SDA1使用配置为I²C模式的MSSP1连接到具有EUI-64™节点标识（24AA02E64T-I/OT）的2K I²C串行EEPROM。
>
> 有关GIPOx器件的更多详细信息，请参见RN2XX3数据手册。

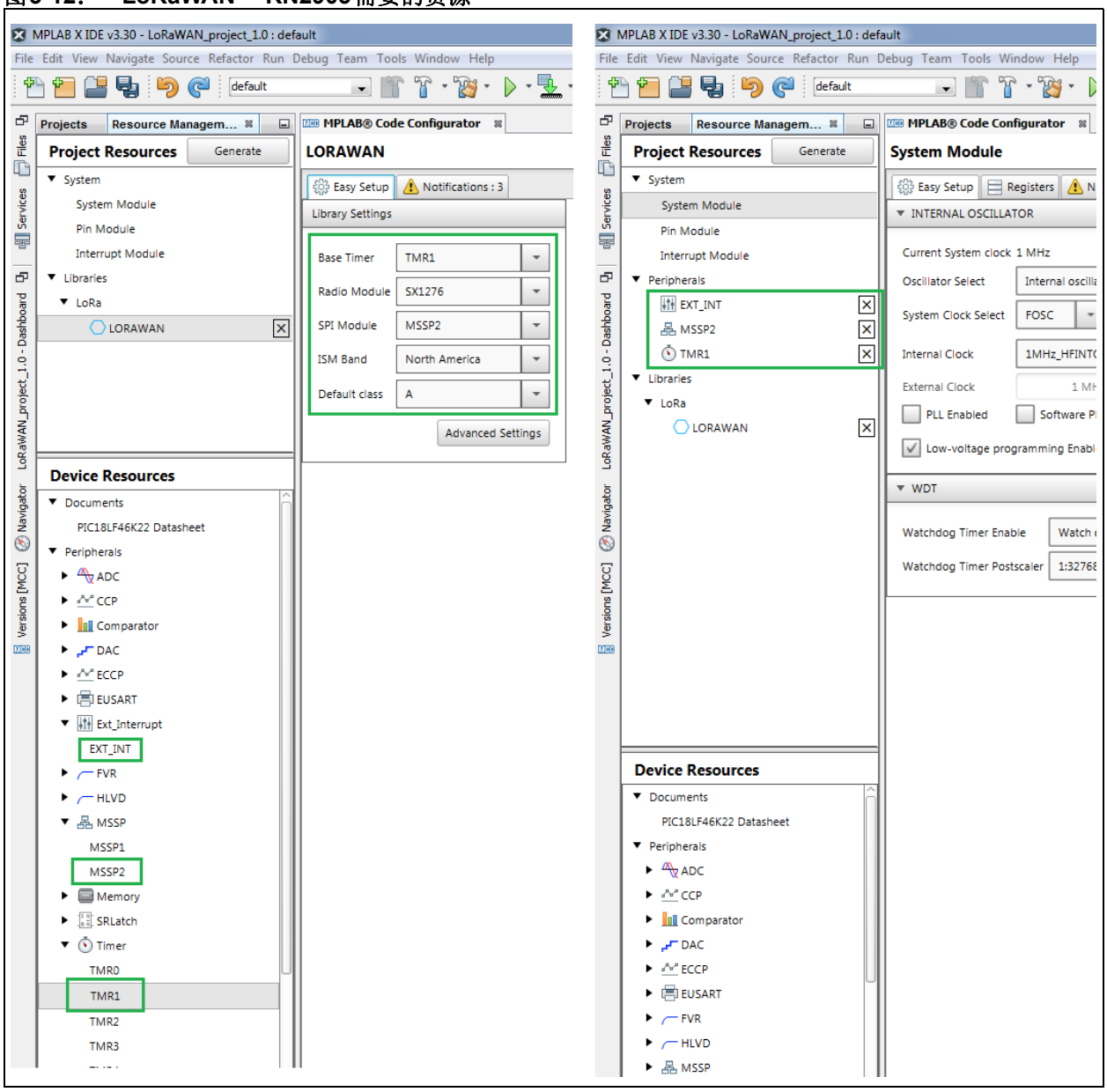
3.2.3.2 向项目添加需要的资源

根据LoRaWAN库在项目的MCC配置期间提供的通知，必须向项目添加以下资源：

- TMR1
- MSSP2
- EXT_INT

用户必须将上述资源添加到Project Resources中，方法是在Device Resources窗口中双击这些项目，如图3-12所示。

图3-12： LoRaWAN™ RN2903需要的资源



© 2017 Microchip Technology Inc.

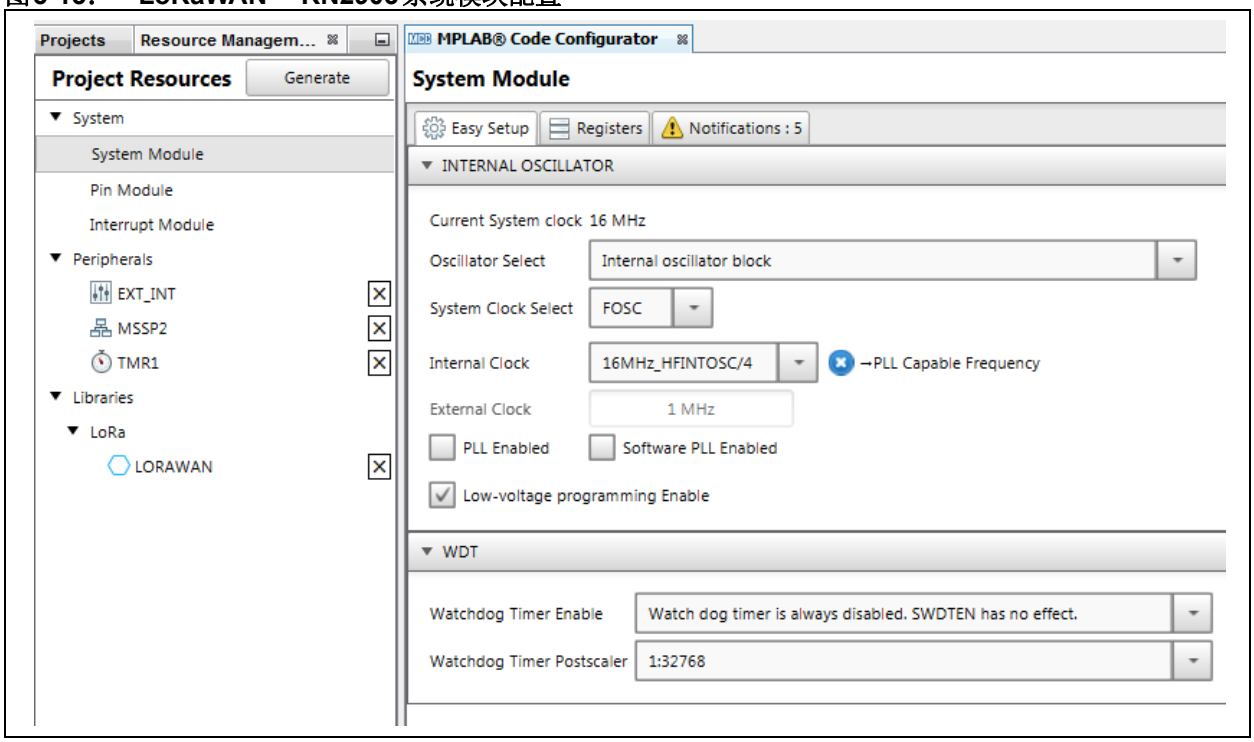
3.2.3.3 LoRaWAN RN2903系统模块配置

用户必须为系统模块设置以下参数（见图3-13）：

- Oscillator Select（振荡器选择）：Internal oscillator block（内部振荡器模块）
- System Clock Select（系统时钟选择）：FOSC
- Internal Clock（内部时钟）：16MHz_HFINTOSC/4
- Low-voltage programming Enable（低电压编程使能）（选中）
- Watchdog Timer Enable（看门狗定时器使能）：Watchdog timer is always disabled. SWDTEN has no effect.（看门狗定时器始终禁止。SWDTEN不起作用。）
- Watchdog Timer Postscaler（看门狗定时器后分频比）：1:32768

启动后，无需对由MCC完成的预加载系统配置进行任何其他修改。

图3-13： LoRaWAN™ RN2903系统模块配置



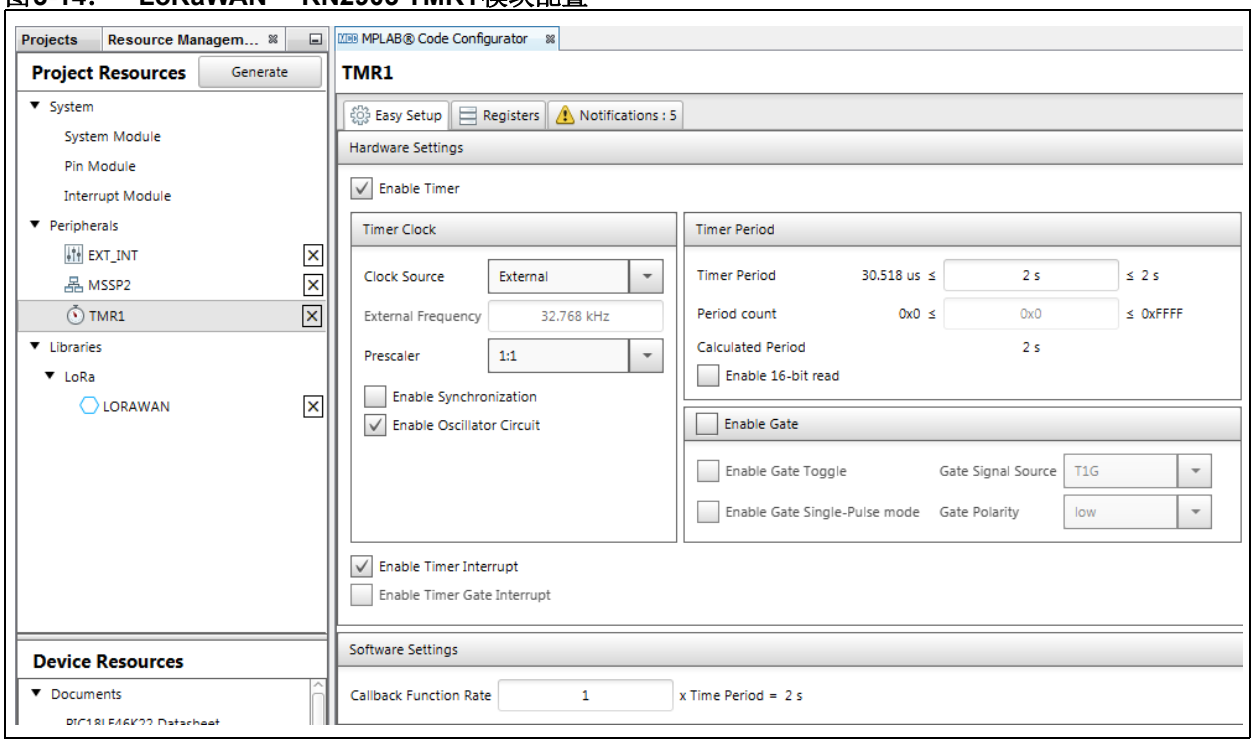
3.2.3.4 LoRaWAN RN2903 TMR1模块配置

用户必须为TMR1模块设置以下参数（见图3-14）：

- Enable Timer（使能定时器）：选中
- Timer Clock（定时器时钟）
  - Clock Source（时钟源）：External（外部）@ 32.768 kHz
  - Prescaler（预分频比）：1:1
  - Enable Synchronization（使能同步）：未选中
  - Enable Oscillator Circuit（使能振荡器电路）：选中
- Timer Period（定时器周期）
  - Timer Period：2s
  - Period Count（周期数）：（由MCC自动填写）
  - Enable 16-bit read（使能16位读操作）：未选中
- Enable Gate（使能门控）：未选中
- Enable Timer Interrupt（允许定时器中断）：选中
- Software Settings（软件设置）
  - Callback Function Rate（回调函数速率）：1

加载TMR1资源后，无需对由MCC完成的预加载系统配置进行任何其他修改。

图3-14： LoRaWAN™ RN2903 TMR1模块配置



© 2017 Microchip Technology Inc.

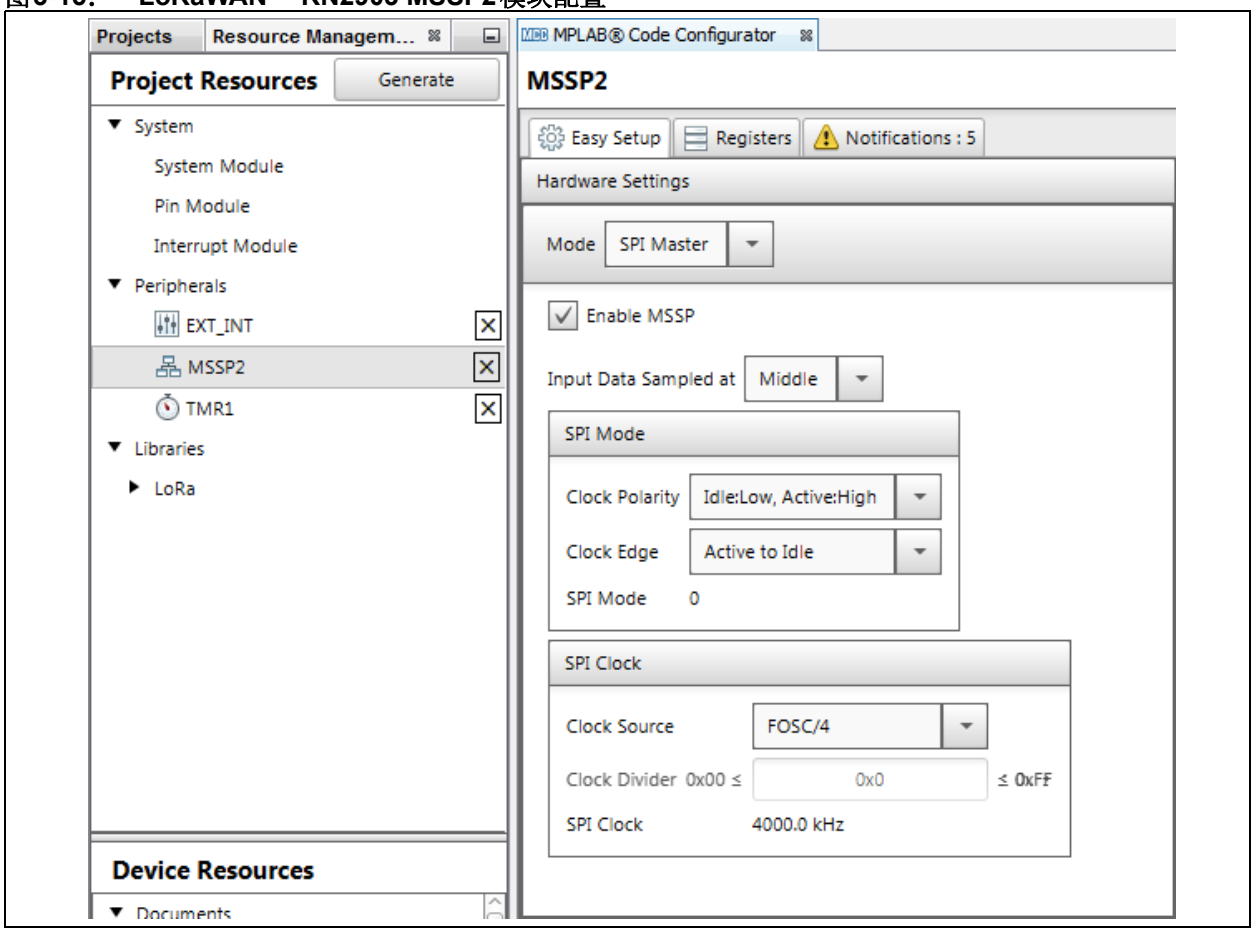
3.2.3.5 LORAWAN RN2903 MSSP2模块配置

用户必须为MSSP2模块设置以下参数（见图3-15）：

- Mode（模式）：SPI Master（SPI主模式）
- Enable MSSP（使能MSSP）：选中
- Input Data Sampled at（输入数据采样点）：Middle（中间）
- SPI Mode（SPI模式）
  - Clock Polarity（时钟极性）：Idle:Low, Active:High（空闲：低电平，工作：高电平）
  - Clock Edge（时钟边沿）：Active to Idle（工作到空闲）
  - SPI Mode：0（由MCC自动填写）
- SPI Clock（SPI时钟）
- Clock Source（时钟源）：FOSC/4
- SPI Clock：4000.0 kHz（由MCC自动填写）

加载MSSP2资源后，无需对由MCC完成的预加载系统配置进行任何其他修改。

图3-15： LoRaWAN™ RN2903 MSSP2模块配置



3.2.3.6 LoRaWAN RN2903 DIO引脚配置

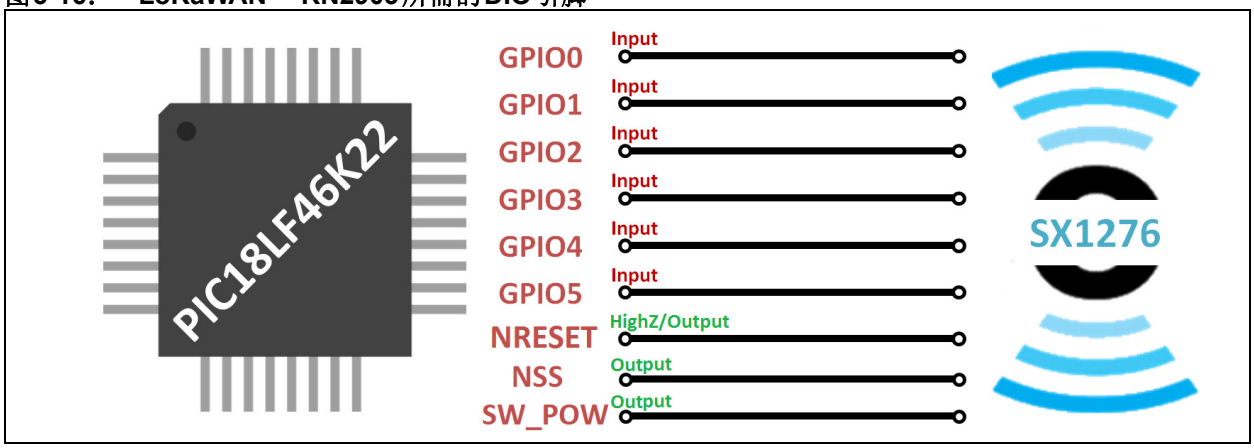
除了SPI，LoRaWAN协议栈还需要6个GPIO（DIO），以便与SX1276无线电收发器进行通信。这些GPIO需要支持中断功能，并且具有外部中断功能或IOC（电平变化中断）功能。

除了这6个GPIO之外，LoRaWAN协议栈还需要使用复位控制引脚（NRESET）和SPI片选（NSS）才能与SX1276无线电收发器和RF开关控制引脚（SW_POW）进行通信。仅当使用RN2903A模块时，才需要RF开关控制引脚。

在“Pin Manager:Grid [MCC]”（引脚管理器：网格[MCC]）窗口中，LoRaWAN库会对可用的MCU引脚进行过滤，仅显示支持中断功能的引脚。

以下配置反映了LoRaWAN DIO引脚在RN2903模块内的连接方式（图3-16）。

图3-16： LoRaWAN™ RN2903所需的DIO引脚



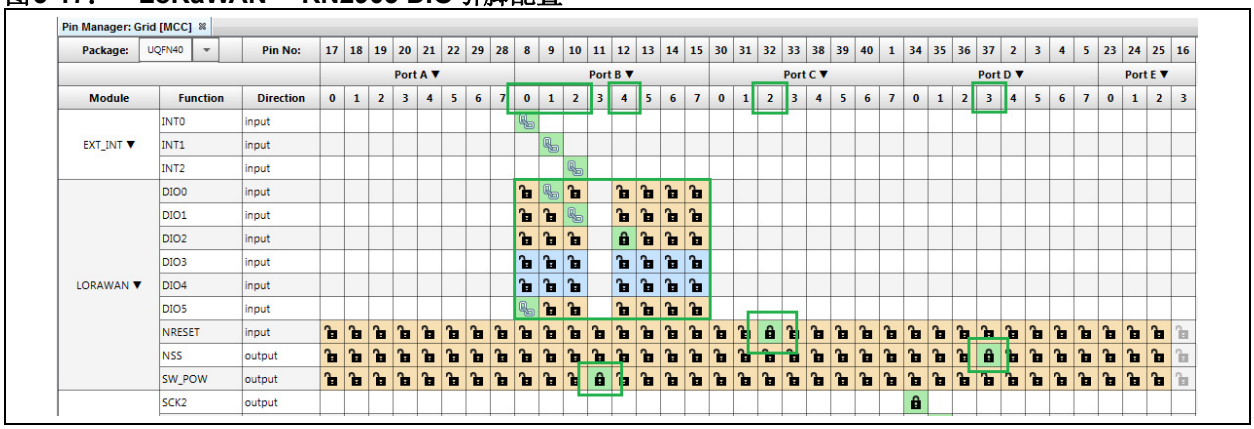
用户必须进行以下选择，如图3-17所示：

- DIO0——PORTB引脚1（RB1）
- DIO1——PORTB引脚2（RB2）
- DIO2——PORTB引脚4（RB4）
- DIO3——不需要（保留供将来使用）
- DIO4——不需要（保留供将来使用）
- DIO5——PORTB引脚0（RB0）
- NRESET——PORTC引脚2（RC2）
- NSS——PORTD引脚3（RD3）
- SW_POW——PORTB引脚3（RB3）。只有RN2903A模块需要该引脚。对于RN2483/RN2903，不需要配置此引脚。

> **注：** 由于SX127X无线电收发器的原因，NRESET引脚需要特殊配置。NRESET引脚的配置顺序如下：首先设置为高阻态，然后设置为输出。更多详细信息，请参见Semtech SX127X数据手册。开发LoRaWAN库时，MCC并不支持高阻态引脚配置。要使NRESET引脚（RC2）实现高阻态配置，必须将其配置为输入。在运行期间，只要不再需要高阻态配置，LoRaWAN协议栈就会将引脚配置为输出。

© 2017 Microchip Technology Inc.

图3-17： LoRaWAN™ RN2903 DIO引脚配置



> **注：** 仅当使用RN2903A模块时，才需要配置SW_POW引脚。如果使用RN2903或RN2483模块，则不需要配置此引脚（如果选择了EU433 MHz或EU868 MHz ISM频段，则SW_POW引脚不会出现在“Pin Manager:Grid [MCC]”窗口中）。

3.2.3.7 LoRaWAN RN2903引脚模块配置

> **注：** 仅当使用RN2903A模块时，才需要配置SW_POW引脚。如果使用RN2903或RN2483模块，则不需要配置此引脚（如果选择了EU433 MHz或EU868 MHz ISM频段，则SW_POW引脚不会出现在“Pin Manager:Grid [MCC]”窗口中）。

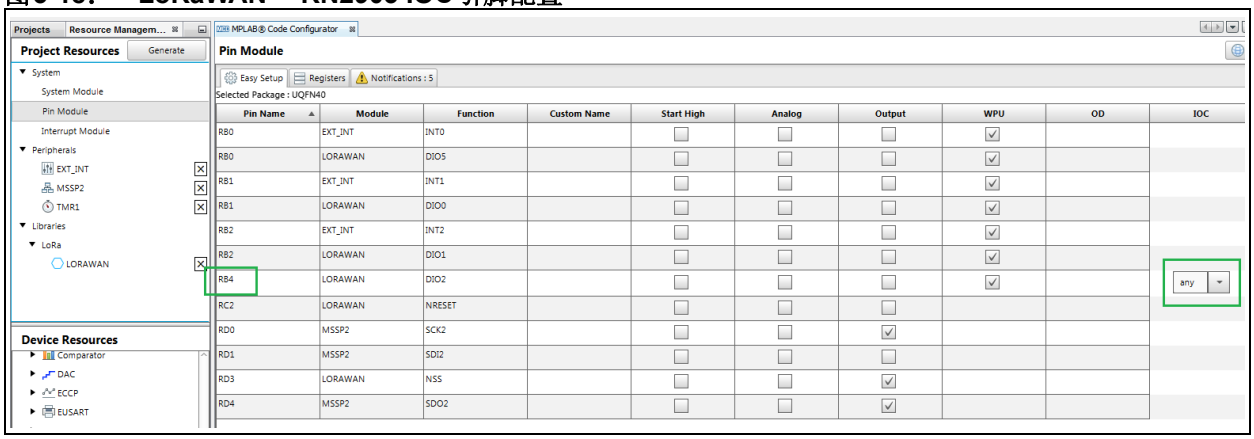
Pin Module（引脚模块）包含添加到项目的资源的引脚。

需要完成以下两处修改：

1. 将RB4的IOC边沿设置为any（任意）。单击Pin Module；该设置位于表的IOC列，请参见图3-18：

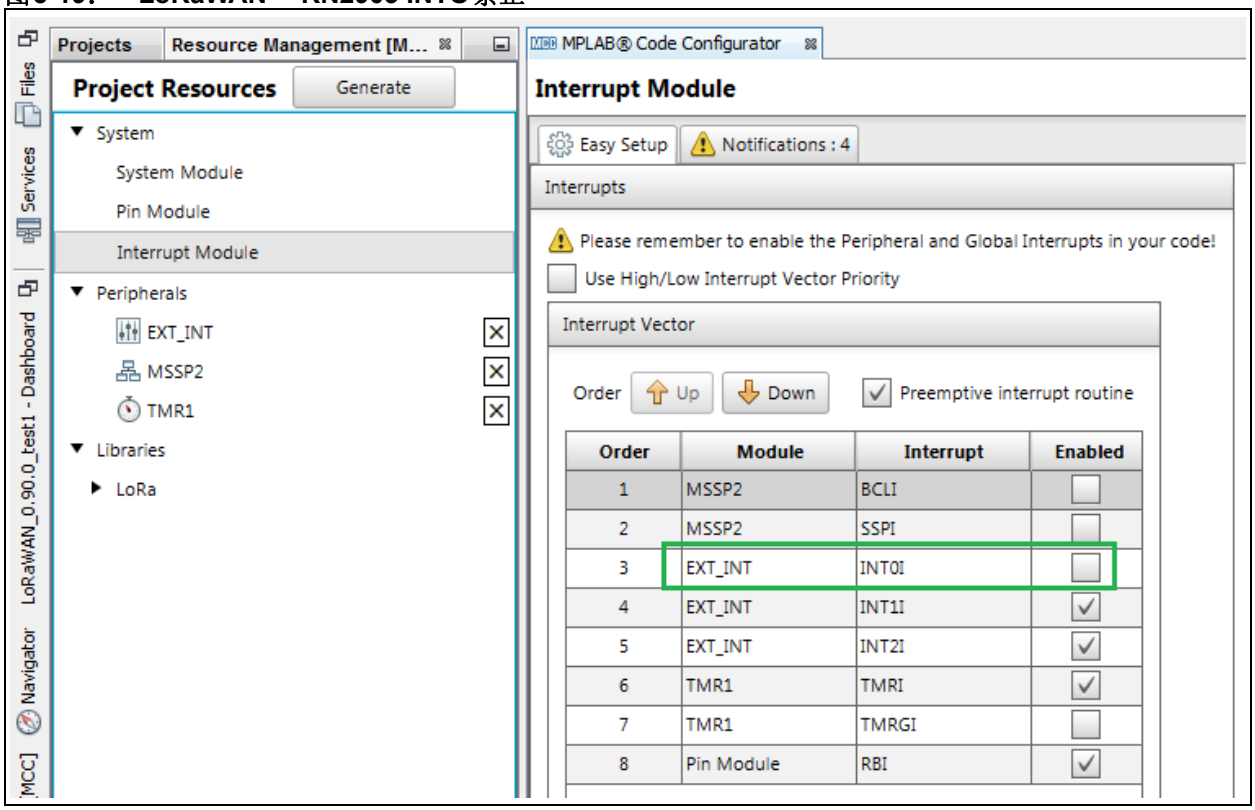
- RB4：将IOC设置为any

图3-18： LoRaWAN™ RN2903 IOC引脚配置



2. 禁止INT0中断。由于对应于INT0中断的RB0引脚只在需要时由LoRaWAN协议栈轮询，因此不需要在INT0上也产生中断。单击**Interrupt Module**（中断模块），清除EXT_INT - INT0I复选框（见图3-19）。

图3-19： LoRaWAN™ RN2903 INTO禁止



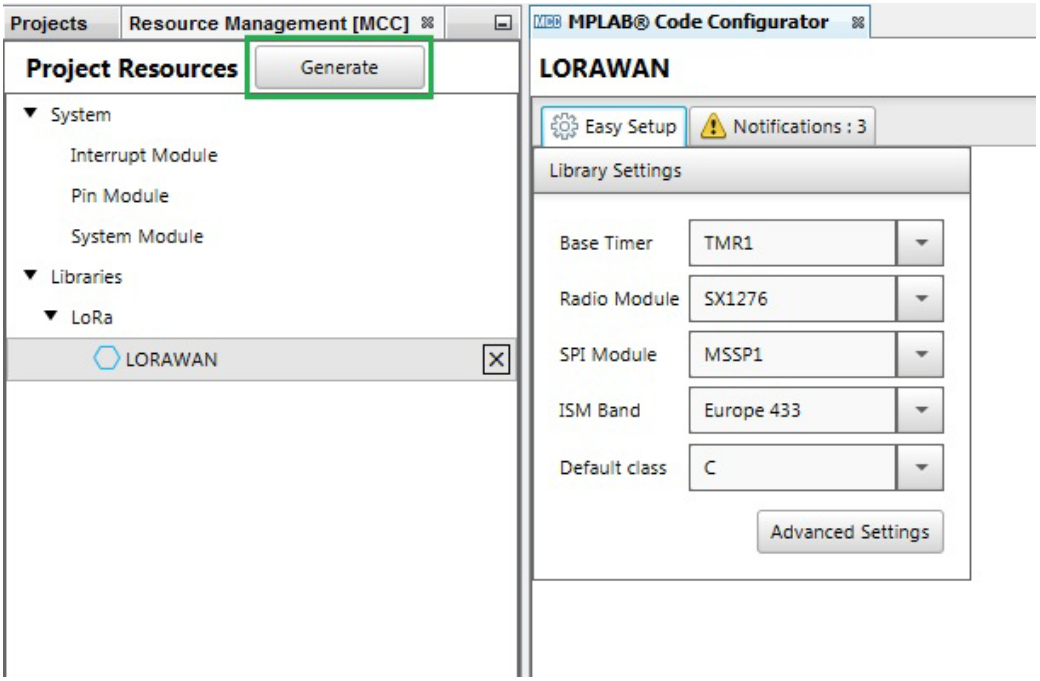
注：	表中的中断顺序可能与图3-19中的中断顺序有所不同。

© 2017 Microchip Technology Inc.

3.3 RN2XX3模块的LoRaWAN™协议栈生成

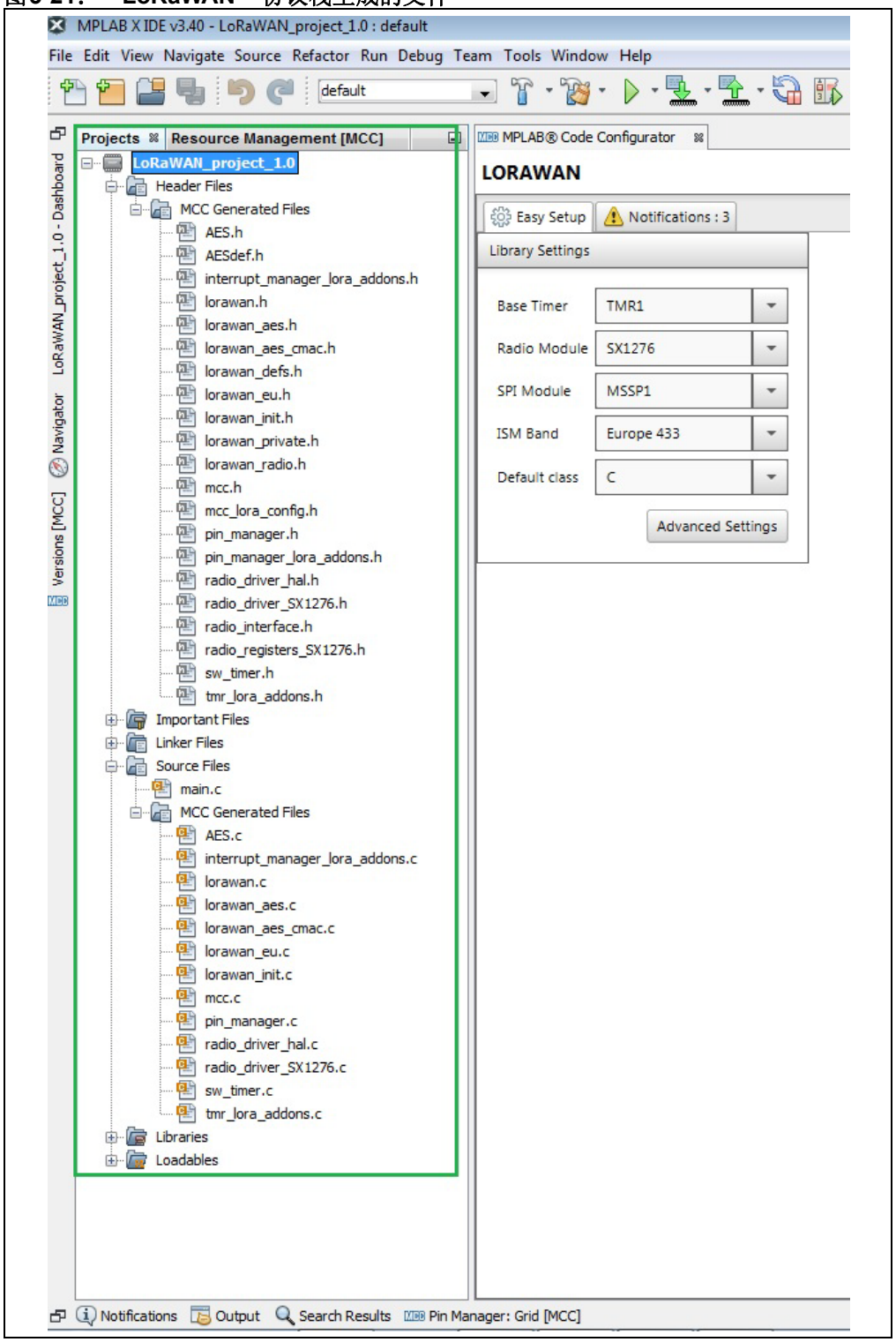
项目配置完成后，可通过按下**Generate**（生成）按钮生成LoRaWAN库，请参见图3-20。

图3-20： RN2903的LoRaWAN™协议栈生成



按下Generate按钮后，即会出现LoRaWAN协议栈，如图3-21所示，用户可以开始创建基于LoRaWAN的应用程序。

图3-21： LoRaWAN™协议栈生成的文件



如果需要使用生成的文件来构建基于LoRaWAN的自定义应用程序，可在**第5章“构建基于LoRaWAN™的应用程序”**中找到更多相关信息。

© 2017 Microchip Technology Inc.

第4章 LoRaWAN™协议栈API

4.1 概述

LoRaWAN协议栈通过MPLAB®代码配置器的图形用户界面来自定义并生成。基于LoRaWAN协议栈的应用程序将使用协议栈提供的API通过LoRa®技术进行数据传输。以下章节解释了LoRaWAN协议栈的应用程序编程接口（Application Programming Interface，API）。

4.2 LoRaWAN架构

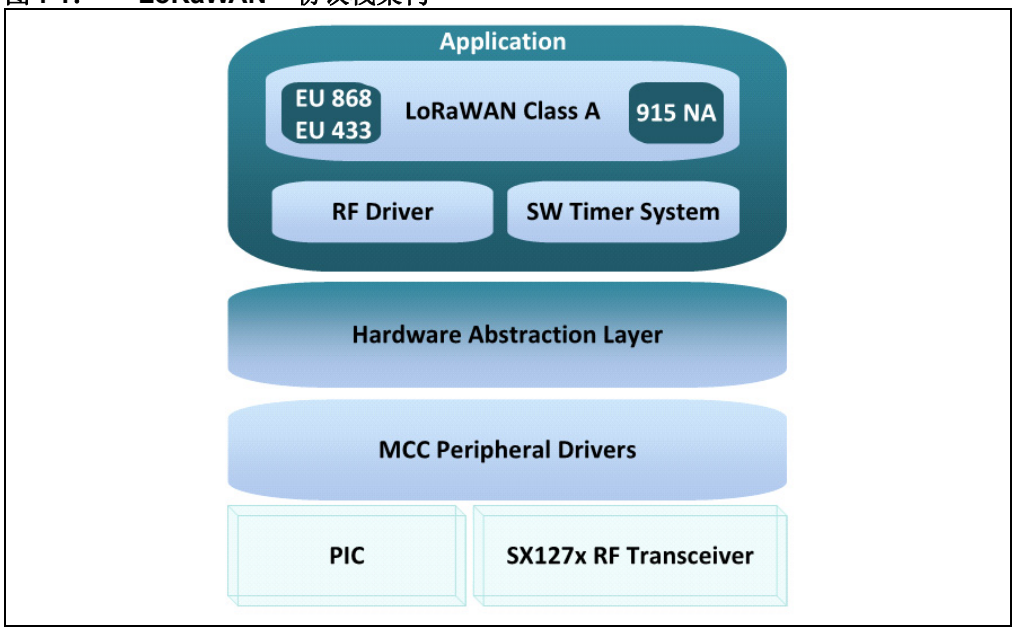
4.2.1 LoRaWAN项目创建

基于LoRaWAN的应用程序分为四层：

1. 应用层，包含用户应用程序（例如，读取传感器和处理数据等）
2. LoRaWAN A类层：
- LoRaWAN A类应用程序（用于北美或欧洲）
- RF驱动程序
- 软件定时器系统
3. 硬件抽象层——在MCC生成的外设驱动程序和应用层之间提供接口
4. MCC外设驱动程序层——包含MCC生成的驱动程序，具体取决于项目配置。

LoRaWAN协议栈和用户自定义应用程序都在RN2XX3模块内部的PIC器件上运行。

图4-1： LoRaWAN™协议栈架构



4.2.2 文件结构

LoRaWAN协议栈文件按照架构层进行拆分。因此，定义和使用以下文件：

1. 用户应用层
   - 用户应用程序头文件

```
• user_custom_header1.h
• user_custom_header2.h
• user_custom_header3.h
```

   - 用户应用程序源文件

```
• main.c
• user_custom_source1.c
• user_custom_source2.c
• user_custom_source3.c
```

2. LoRaWAN协议栈
   - LoRaWAN头文件

```
• interrupt_manager_lora_addons.h
• lorawan.h
• lorawan_aes.h
• lorawan_aes_cmac.h
• lorawan_defs.h
• lorawan_init.h
• lorawan_na.h
• lorawan_eu.h
• lorawan_private.h
• lorawan_radio.h
• AES.h
• AESdef.h
```

   - LoRaWAN源文件

```
• AES.c
• interrupt_manager_lora_addons.c
• lorawan.c
• lorawan_aes.c
• lorawan_aes_cmac.c
• lorawan_init.c
• lorawan_na.c
• lorawan_eu.c
```


© 2017 Microchip Technology Inc.

- 无线电驱动程序头文件

```
• radio_driver_SX1276.h
• radio_driver_SX1272.h
• radio_interface.h
• radio_registers_SX1276.h
• radio_registers_SX1272.h
```

- 无线电驱动程序源文件

```
• radio_driver_SX1276.h
• radio_driver_SX1272.h
```

- 软件定时器系统头文件

```
• sw_timer.h
• tmr_lora_addons.h
```

- 软件定时器系统源文件

```
• sw_timer.c
• tmr_lora_addons.c
```

3. 硬件抽象层（HAL文件）
   - HAL头文件

```
• mcc_lora_config.h
• tmr_lora_addons.h
• pin_manager_lora_addons.h
• radio_driver_hal.h
```

   - HAL源文件

```
• radio_driver_hal.c
```

4. MCC外设驱动程序：除了库外，还包括MCC为使用的每个MCU模块生成的所有文件（例如：SPI2——`spi2.h`和`spi2.c`；TMR1——`tmr1.h`和`tmr1.c`）。

4.3 应用程序编程接口

4.3.1 概述

LoRaWAN协议栈提供用户在构建基于LoRaWAN的自定义应用程序的过程中所需的所有应用程序编程接口（API）。

API在lorawan.c文件中定义，并在lorawan.h文件中声明（附有所有详细信息）。下一节提供了有关API的详细信息。

4.3.2 LoRaWAN API

4.3.2.1 LORAWAN_Init

名称	LORAWAN_Init
原型	void LORAWAN_Init(RxAppDataCb_t RxPayload, RxJoinResponseCb_t RxJoinResponse)
摘要	LoRaWAN初始化函数
说明	初始化LoRaWAN协议栈和无线电模块。
前提条件	无
参数	• RxPayload——指向在双向通信结束后调用的函数的指针 • RxJoinResponse——指向在激活过程后调用的函数的指针
返回	无
示例	LORAWAN_Init(RxData, RxJoinResponse);

4.3.2.2 LORAWAN_Join

名称	LORAWAN_Join
原型	LorawanError_t LORAWAN_Join(ActivationType_t activationTypeNew)
摘要	LoRaWAN激活过程
说明	此函数用于启动LoRaWAN激活过程。每当调用注册的RxJoinResponse回调时，便会完成此过程。
前提条件	无
参数	• activationTypeNew——激活类型：OTAA或ABP
返回	函数返回操作的状态（LorawanError_t）
示例	LORAWAN_Join(ABP);

© 2017 Microchip Technology Inc.

4.3.2.3 LORAWAN_Send

名称	LORAWAN_Send
原型	LorawanError_t LORAWAN_Send (TransmissionType_t confirmed, uint8_t port, void *buffer, uint8_t bufferLength)
摘要	双向通信启动
说明	此函数用于启动双向通信过程。每当调用注册的RxPayload回调时，便会完成此过程。
前提条件	无
参数	• confirmed——表示传输类型；可以是UNCNF（未确认）或CNF（已确认）（TransmissionType_t） • port——表示正在进行传输的端口；它是一个介于0和255之间的数字（uint8_t） • buffer——用于存储要发送的数据的数据缓冲区 • bufferLength——数据缓冲区的长度，以字节为单位（uint8_t）
返回	函数返回操作的状态（LorawanError_t）
示例	LORAWAN_Send(UNCNF, 4, “Hello World!”, 12);

4.3.2.4 LORAWAN_SetDeviceEui

名称	LORAWAN_SetDeviceEui
原型	void LORAWAN_SetDeviceEui (uint8_t *deviceEuiNew)
摘要	设置终端设备标识符的值。
说明	此函数用于设置终端设备标识符（DevEUI）。 DevEUI是惟一标识终端设备的IEEE EUI64地址空间中的全局终端设备ID。
前提条件	指针必须由调用程序分配。
参数	• deviceEui——将存储值的缓冲区。
返回	无
示例	uint8_t deviceEui[8] = {0x00, 0x04, 0xA3, 0x0B, 0x00, 0x1A, 0x9E, 0xF8}; LORAWAN_SetDeviceEui(&deviceEui[0]);

4.3.2.5 LORAWAN_GetDeviceEui

名称	LORAWAN_GetDeviceEui
原型	void LORAWAN_GetDeviceEui (uint8_t *deviceEui)
摘要	获取终端设备标识符的值。
说明	此函数用于获取终端设备标识符（DevEUI）。 DevEUI是惟一标识终端设备的IEEE EUI64地址空间中的全局终端设备ID。
前提条件	指针必须由调用程序分配。
参数	• deviceEui——将存储值的缓冲区。
返回	无
示例	uint8_t deviceEui[8]; LORAWAN_GetDeviceEui(&deviceEui[0]);

4.3.2.6 LORAWAN_SetApplicationEui

名称	LORAWAN_SetApplicationEui
原型	void LORAWAN_SetApplicationEui (uint8_t *applicationEuiNew)
摘要	设置应用程序标识符。
说明	此函数用于设置终端设备应用程序标识符（AppEUI）。 AppEUI是惟一标识终端设备应用程序提供方（即所有者）的IEEE EUI64地址空间中的全局应用程序ID。
前提条件	指针必须由调用程序分配。
参数	• applicationEuiNew——存储AppEUI的缓冲区。
返回	无
示例	uint8_t applicationEuiNew[8] = {0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88}; LORAWAN_SetApplicationEui(&applicationEuiNew[0]);

4.3.2.7 LORAWAN_GetApplicationEui

名称	LORAWAN_GetApplicationEui
原型	void LORAWAN_GetApplicationEui (uint8_t *applicationEui)
摘要	获取应用程序标识符的值。
说明	此函数用于获取终端设备应用程序标识符（AppEUI）。 AppEUI是惟一标识终端设备应用程序提供方（即所有者）的IEEE EUI64地址空间中的全局应用程序ID。
前提条件	指针必须由调用程序分配。
参数	• applicationEui——将存储值的缓冲区。
返回	无
示例	uint8_t applicationEui[8]; LORAWAN_GetApplicationEui(&applicationEui[0]);

4.3.2.8 LORAWAN_SetDeviceAddress

名称	LORAWAN_SetDeviceAddress
原型	void LORAWAN_SetDeviceAddress (uint32_t deviceAddressNew)
摘要	设置终端设备地址。
说明	此函数用于设置终端设备地址（DevAddr）。DevAddr是当前网络中的终端设备的32位标识符。
前提条件	无
参数	• deviceAddressNew——要设置的新地址的值。
返回	无
示例	uint32_t devAddr = 0x11223344; LORAWAN_SetDeviceAddress(devAddr);

© 2017 Microchip Technology Inc.

4.3.2.9 LORAWAN_GetDeviceAddress

名称	LORAWAN_GetDeviceAddress
原型	uint32_t LORAWAN_GetDeviceAddress (void)
摘要	返回终端设备地址。
说明	此函数用于获取终端设备地址（DevAddr）。DevAddr是当前网络中的终端设备的32位标识符。
前提条件	无
参数	无
返回	设备地址（uint32_t值）
示例	uint32_t devAddr; devAddr = LORAWAN_GetDeviceAddress();

4.3.2.10 LORAWAN_SetNetworkSessionKey

名称	LORAWAN_SetNetworkSessionKey
原型	void LORAWAN_SetNetworkSessionKey (uint8_t *networkSessionKeyNew)
摘要	设置网络会话密钥。
说明	此函数用于设置网络会话密钥（NwkSKey）。 NwkSKey是特定于终端设备的网络会话密钥，用于计算和验证MIC（消息完整性代码）。 NwkSKey是一个16字节的数组。
前提条件	指针必须由调用程序分配。
参数	• networkSessionKeyNew——将存储值的缓冲区。
返回	无
示例	uint8_t nwkSKey[16] = {0x2B, 0x7E, 0x15, 0x16, 0x28, 0xAE, 0xD2, 0xA6, 0xAB, 0xF7, 0x15, 0x88, 0x09, 0xCF, 0x4F, 0x3C}; LORAWAN_SetNetworkSessionKey(nwkSKey);

4.3.2.11 LORAWAN_GetNetworkSessionKey

名称	LORAWAN_GetNetworkSessionKey
原型	void LORAWAN_GetNetworkSessionKey (uint8_t *networkSessionKey)
摘要	获取网络会话密钥。
说明	此函数用于获取网络会话密钥（NwkSKey）。 NwkSKey是特定于终端设备的网络会话密钥，用于计算和验证MIC（消息完整性代码）。 NwkSKey是一个16字节的数组。
前提条件	指针必须由调用程序分配。
参数	• networkSessionKey——将存储值的缓冲区。
返回	无
示例	uint8_t nwkSKey[16]; LORAWAN_GetNetworkSessionKey(nwkSKey);

4.3.2.12 LORAWAN_SetApplicationSessionKey

名称	LORAWAN_SetApplicationSessionKey
原型	void LORAWAN_SetApplicationSessionKey (uint8_t *applicationSessionKeyNew)
摘要	设置应用程序会话密钥。
说明	此函数用于设置应用程序会话密钥（AppSKey）。 AppSKey是特定于终端设备的应用程序会话密钥，用于对应用程序特定的数据消息的有效负载字段进行加密/解密，还用于计算/验证有效负载中可能包含的应用程序级MIC（消息完整性代码）。 AppSKey是一个16字节的数组。
前提条件	指针必须由调用程序分配。
参数	• applicationSessionKeyNew——存储值的缓冲区。
返回	无
示例	uint8_t appSKey[16] = {0x3C, 0x8F, 0x26, 0x27, 0x39, 0xBF, 0xE3, 0xB7, 0xBC, 0x08, 0x26, 0x99, 0x1A, 0xD0, 0x50, 0x4D}; LORAWAN_SetApplicationSessionKey(appSKey);

© 2017 Microchip Technology Inc.

4.3.2.13 LORAWAN_GetApplicationSessionKey

名称	LORAWAN_GetApplicationSessionKey
原型	void LORAWAN_GetApplicationSessionKey (uint8_t *applicationSessionKey)
摘要	获取应用程序会话密钥。
说明	此函数用于获取应用程序会话密钥（AppSKey）。 AppSKey是特定于终端设备的应用程序会话密钥，用于对应用程序特定的数据消息的有效负载字段进行加密/解密，还用于计算/验证有效负载中可能包含的应用程序级MIC（消息完整性代码）。 AppSKey是一个16字节的数组。
前提条件	指针必须由调用程序分配。
参数	• applicationSessionKey——将存储值的缓冲区。
返回	无
示例	uint8_t appSKey[16]; LORAWAN_GetApplicationSessionKey(appSKey);

4.3.2.14 LORAWAN_SetApplicationKey

名称	LORAWAN_SetApplicationKey
原型	void LORAWAN_SetApplicationKey (uint8_t *applicationKeyNew)
摘要	设置应用程序密钥。
说明	此函数用于设置应用程序会话密钥（AppKey）。 AppKey是特定于终端设备的AES-128应用程序密钥，由应用程序所有者分配给终端设备。
前提条件	指针必须由调用程序分配。
参数	• applicationKeyNew——存储值的缓冲区。
返回	无
示例	uint8_t appKey[16] = {0x3C, 0x8F, 0x26, 0x27, 0x39, 0xBF, 0xE3, 0xB7, 0xBC, 0x08, 0x26, 0x99, 0x1A, 0xD0, 0x50, 0x4D}; LORAWAN_SetApplicationKey(appKey);

4.3.2.15 LORAWAN_GetApplicationKey

名称	LORAWAN_GetApplicationKey
原型	void LORAWAN_GetApplicationKey (uint8_t *applicationKey)
摘要	获取应用程序密钥。
说明	此函数用于获取应用程序会话密钥（AppKey）。 AppKey是特定于终端设备的AES-128应用程序密钥，由应用程序所有者分配给终端设备。
前提条件	指针必须由调用程序分配。
参数	• applicationKey——存储值的缓冲区。
返回	无
示例	uint8_t appKey[16]; LORAWAN_GetApplicationKey(appKey);

4.3.2.16 LORAWAN_SetAdr

名称	LORAWAN_SetAdr
原型	void LORAWAN_SetAdr (bool status)
摘要	设置自适应数据速率模式。
说明	此函数用于设置自适应数据速率（Adaptive Data Rate，ADR）模式。 LoRa网络允许终端设备单独使用任何可能的数据速率，即所谓的自适应数据速率（ADR）。
前提条件	无
参数	status——真/假
返回	无
示例	LORAWAN_SetAdr (true);

© 2017 Microchip Technology Inc.

4.3.2.17 LORAWAN_GetAdr

名称	LORAWAN_GetAdr
原型	bool LORAWAN_GetAdr (void)
摘要	返回自适应数据速率模式。
说明	此函数用于返回自适应数据速率（ADR）模式。 LoRa网络允许终端设备单独使用任何可能的数据速率，即所谓的自适应数据速率（ADR）。 如果设置了ADR，网络将通过适当的MAC命令控制终端设备的数据速率。 如果未设置ADR，则无论接收的信号质量如何，网络都不会尝试控制终端设备的数据速率。
前提条件	无
参数	无
返回	status——真/假
示例	bool adrStatus; adrStatus = LORAWAN_GetAdr();

4.3.2.18 LORAWAN_GetCurrentDataRate

名称	LORAWAN_GetCurrentDataRate
原型	uint8_t LORAWAN_GetCurrentDataRate (void)
摘要	返回下一次上行链路传输的数据速率模式。
说明	终端设备和网关之间的通信在不同的频率通道上以不同数据速率传播。 数据速率的选择是针对通信范围与消息持续时间的折衷；但不同数据速率的通信之间不会彼此干扰。
前提条件	无
参数	• valueNew——新的数据速率值
返回	返回当前数据速率（uint8_t）
示例	uint8_t dataRateUsed; dataRateUsed = LORAWAN_GetCurrentDataRate();

4.3.2.19 LORAWAN_SetTxPower

名称	LORAWAN_SetTxPower
原型	LorawanError_t LORAWAN_SetTxPower (uint8_t txPowerNew)
摘要	设置TX输出功率。
说明	TX输出功率（TXPower）特定于区域。txPowerNew必须作为索引提供，其值介于0与15之间。 更多详细信息，请参见LoRaWAN规范V1.0文档。
前提条件	无
参数	• txPowerNew——新的TX功率值
返回	返回LoRaWAN错误类型（LorawanError_t）
示例	LORAWAN_SetTxPower (4);

4.3.2.20 LORAWAN_GetTxPower

名称	LORAWAN_GetTxPower
原型	uint8_t LORAWAN_GetTxPower (void)
摘要	返回TX输出功率。
说明	TX输出功率（TXPower）特定于区域。txPower作为索引返回，其值介于0与15之间。 更多详细信息，请参见LoRaWAN规范V1.0文档。
前提条件	无
参数	• txPowerNew——新的TX功率值
返回	返回LoRaWAN错误类型（LorawanError_t）
示例	uint8_t txPowerUsed; txPowerUsed = LORAWAN_GetTxPower();

4.3.2.21 LORAWAN_SetSyncWord

名称	LORAWAN_SetSyncWord
原型	void LORAWAN_SetSyncWord (uint8_t syncWord)
摘要	设置同步字。
说明	此函数用于设置通信期间使用的当前同步字。 更多详细信息，请参见LoRaWAN规范V1.0文档。
前提条件	无
参数	• syncWord——新同步字的值
返回	无
示例	uint8_t syncWordToSet = 0x34; LORAWAN_SetSyncWord(syncWordToSet);

© 2017 Microchip Technology Inc.

4.3.2.22 LORAWAN_GetSyncWord

名称	LORAWAN_GetSyncWord
原型	uint8_t LORAWAN_GetSyncWord (void)
摘要	返回同步字。
说明	此函数用于返回通信期间使用的当前同步字。 更多详细信息，请参见LoRaWAN规范V1.0文档。
前提条件	无
参数	无
返回	同步字的值（uint8_t）
示例	uint8_t syncWordUsed; syncWordUsed = LORAWAN_SetSyncWord();

4.3.2.23 LORAWAN_SetUplinkCounter

名称	LORAWAN_SetUplinkCounter
原型	void LORAWAN_SetUplinkCounter (uint32_t ctr)
摘要	设置当前上行链路计数器。
说明	此函数用于设置通信期间使用的当前上行链路计数器。 这可以用于使上行链路计数器与由服务器存储的值同步。
前提条件	无
参数	• ctr——要设置的新计数器的值。
返回	无
示例	LORAWAN_SetUplinkCounter(1000);

4.3.2.24 LORAWAN_GetUplinkCounter

名称	LORAWAN_GetUplinkCounter
原型	uint32_t LORAWAN_GetUplinkCounter (void)
摘要	返回当前上行链路计数器。
说明	此函数用于返回通信期间使用的当前上行链路计数器。 这可以用于使上行链路计数器与由服务器存储的值同步。
前提条件	无
参数	当前上行链路计数器（uint32_t）
返回	无
示例	uint32_t uplinkCntUsed; LORAWAN_GetUplinkCounter(uplinkCntUsed);

4.3.2.25 LORAWAN_SetDownlinkCounter

名称	LORAWAN_SetDownlinkCounter
原型	void LORAWAN_SetDownlinkCounter (uint32_t ctr)
摘要	设置当前下行链路计数器。
说明	此函数用于设置通信期间使用的当前下行链路计数器。 这可以用于使下行链路计数器与由服务器存储的值同步。
前提条件	无
参数	• ctr——新计数器的值
返回	无
示例	LORAWAN_SetDownlinkCounter(1500);

4.3.2.26 LORAWAN_GetDownlinkCounter

名称	LORAWAN_GetDownlinkCounter
原型	uint32_t LORAWAN_GetDownlinkCounter (void)
摘要	返回当前下行链路计数器。
说明	此函数用于返回通信期间使用的当前下行链路计数器。 这可以用于使下行链路计数器与由服务器存储的值同步。
前提条件	无
参数	当前下行链路计数器（uint32_t）
返回	无
示例	uint32_t downlinkCntUsed; LORAWAN_GetDownlinkCounter(downlinkCntUsed);

4.3.2.27 LORAWAN_SetReceiveDelay1

名称	LORAWAN_SetReceiveDelay1
原型	void LORAWAN_SetReceiveDelay1 (uint16_t receiveDelay1New)
摘要	设置第一个接收延时（RECEIVE_DELAY1）的值。
说明	此函数将设置发送和第一个接收窗口之间的延时。 发送和第二个接收窗口之间的延时在软件中计算，具体为发送和第一个接收窗口之间的延时 + 1000（单位为毫秒）。
前提条件	无
参数	• receiveDelay1New——新延时的值（必须以毫秒为单位提供）。
返回	无
示例	LORAWAN_SetReceiveDelay1(1100);

© 2017 Microchip Technology Inc.

4.3.2.28 LORAWAN_GetReceiveDelay1

名称	LORAWAN_GetReceiveDelay1
原型	uint16_t LORAWAN_GetReceiveDelay1 (void)
摘要	返回第一个接收延时（RECEIVE_DELAY1）的值。
说明	此函数将返回发送和第一个接收窗口之间的延时。 发送和第二个接收窗口之间的延时在软件中计算，具体为发送和第一个接收窗口之间的延时 + 1000（单位为毫秒）。
前提条件	无
参数	无
返回	接收延时的值（以毫秒返回——uint16_t）
示例	uint16_t receiveDelay1Used; receiveDelay1Used = LORAWAN_GetReceiveDelay1();

4.3.2.29 LORAWAN_GetReceiveDelay2

名称	LORAWAN_GetReceiveDelay2
原型	uint16_t LORAWAN_GetReceiveDelay2 (void)
摘要	返回第二个接收延时（RECEIVE_DELAY2）的值。
说明	此函数将返回发送和第二个接收窗口之间的延时。 发送和第二个接收窗口之间的延时在软件中计算，具体为发送和第一个接收窗口之间的延时 + 1000（单位为毫秒）。
前提条件	无
参数	无
返回	第二个接收延时的值（以毫秒返回——uint16_t）
示例	uint16_t receiveDelay2Used; receiveDelay2Used = LORAWAN_GetReceiveDelay2();

4.3.2.30 LORAWAN_SetJoinAcceptDelay1

名称	LORAWAN_SetJoinAcceptDelay1
原型	void LORAWAN_SetJoinAcceptDelay1 (uint16_t joinAcceptDelay1New)
摘要	设置第一个加入接受延时（JOIN_ACCEPT_DELAY1）的值。
说明	如果允许终端设备加入网络，则网络服务器将用join-accept消息来响应join-request消息。 join-accept消息的发送方式与普通下行链路类似，只是使用延时JOIN_ACCEPT_DELAY1或JOIN_ACCEPT_DELAY2（而不分别使用RECEIVE_DELAY1和RECEIVE_DELAY2）。
前提条件	无
参数	• joinAcceptDelay1New——新加入接受延时的值（必须以毫秒为单位提供）
返回	无
示例	LORAWAN_SetJoinAcceptDelay1(5500);

4.3.2.31 LORAWAN_GetJoinAcceptDelay1

名称	LORAWAN_GetJoinAcceptDelay1
原型	uint16_t LORAWAN_GetJoinAcceptDelay1 (void)
摘要	返回第一个加入接受延时（JOIN_ACCEPT_DELAY1）的值。
说明	如果允许终端设备加入网络，则网络服务器将用join-accept消息来响应join-request消息。 join-accept消息的发送方式与普通下行链路类似，只是使用延时JOIN_ACCEPT_DELAY1或JOIN_ACCEPT_DELAY2（而不分别使用RECEIVE_DELAY1和RECEIVE_DELAY2）。
前提条件	无
参数	无
返回	第一个加入接受延时的值（将以毫秒返回——uint16_t）
示例	uint16_t joinAcceptDelay1Used; joinAcceptDelay1Used = LORAWAN_GetJoinAcceptDelay1();

© 2017 Microchip Technology Inc.

4.3.2.32 LORAWAN_SetJoinAcceptDelay2

名称	LORAWAN_SetJoinAcceptDelay2
原型	void LORAWAN_SetJoinAcceptDelay2 (uint16_t joinAcceptDelay2New)
摘要	设置第二个加入接受延时（JOIN_ACCEPT_DELAY2）的值。
说明	如果允许终端设备加入网络，则网络服务器将用join-accept消息来响应join-request消息。 join-accept消息的发送方式与普通下行链路类似，只是使用延时JOIN_ACCEPT_DELAY1或JOIN_ACCEPT_DELAY2（而不分别使用RECEIVE_DELAY1和RECEIVE_DELAY2）。
前提条件	无
参数	• joinAcceptDelay2New——新加入接受延时的值（必须以毫秒为单位提供）
返回	无
示例	LORAWAN_SetJoinAcceptDelay2(6500);

4.3.2.33 LORAWAN_GetJoinAcceptDelay2

名称	LORAWAN_GetJoinAcceptDelay2
原型	uint16_t LORAWAN_GetJoinAcceptDelay2 (void)
摘要	返回第二个加入接受延时（JOIN_ACCEPT_DELAY2）的值。
说明	如果允许终端设备加入网络，则网络服务器将用join-accept消息来响应join-request消息。 join-accept消息的发送方式与普通下行链路类似，只是使用延时JOIN_ACCEPT_DELAY1或JOIN_ACCEPT_DELAY2（而不分别使用RECEIVE_DELAY1和RECEIVE_DELAY2）。
前提条件	无
参数	无
返回	第一个加入接受延时的值（将以毫秒返回——uint16_t）
示例	uint16_t joinAcceptDelay2Used; joinAcceptDelay2Used = LORAWAN_GetJoinAcceptDelay2();

4.3.2.34 LORAWAN_SetMaxFcntGap

名称	LORAWAN_SetMaxFcntGap
原型	void LORAWAN_SetMaxFcntGap (uint16_t maxFcntGapNew)
摘要	设置最大帧计数器间隙（MAX_FCNT_GAP）的值。
说明	每个终端设备具有两个帧计数器，用于跟踪上行链路发送到网络服务器的数据帧的数量（FCntUp，由终端设备递增计数）以及终端设备下行链路从网络服务器接收的数据帧的数量（FCntDown，由网络服务器递增计数）。 在接收器侧，对应的计数器与接收到的值保持同步，前提是接收到的值相对于当前计数器值已递增，并且在考虑计数器计满返回之后低于由MAX_FCNT_GAP指定的值。 如果此差值大于MAX_FCNT_GAP的值，则表示已丢失过多数据帧，并且随后会将其丢弃。
前提条件	无
参数	• maxFcntGapNew——新的最大帧计数器的值
返回	无
示例	LORAWAN_SetMaxFcntGap(3000);

4.3.2.35 LORAWAN_GetMaxFcntGap

名称	LORAWAN_GetMaxFcntGap
原型	uint16_t LORAWAN_GetMaxFcntGap (void)
摘要	设置自适应数据速率确认限制（ADR_ACK_LIMIT）的值。
说明	每次上行链路帧计数器递增（对于每个新的上行链路，重复传输不会增加计数器的值）时，器件都会使ADR_ACK_CNT计数器递增。 在没有任何下行链路响应的ADR_ACK_LIMIT上行链路之后（ADR_ACK_CNT >= ADR_ACK_LIMIT），它会将ADR确认请求位（ADRACKReq）置1。 网络需要在接下来的ADR_ACK_DELAY帧内用下行链路帧进行响应，在上行链路帧之后接收的任何下行链路帧都将复位ADR_ACK_CNT计数器。
前提条件	无
参数	无
返回	最大帧计数器的值
示例	uint16_t maxFcntGapUsed; maxFcntGapUsed = LORAWAN_GetMaxFcntGap();

© 2017 Microchip Technology Inc.

4.3.2.36 LORAWAN_SetAdrAckLimit

名称	LORAWAN_SetAdrAckLimit
原型	void LORAWAN_SetAdrAckLimit (uint8_t adrAckLimitNew)
摘要	设置自适应数据速率确认限制（ADR_ACK_LIMIT）的值。
说明	每次上行链路帧计数器递增（对于每个新的上行链路，重复传输不会增加计数器的值）时，器件都会使ADR_ACK_CNT计数器递增。 在没有任何下行链路响应的ADR_ACK_LIMIT上行链路之后（ADR_ACK_CNT >= ADR_ACK_LIMIT），它会将ADR确认请求位（ADRACKReq）置1。 网络需要在接下来的ADR_ACK_DELAY帧内用下行链路帧进行响应，在上行链路帧之后接收的任何下行链路帧都将复位ADR_ACK_CNT计数器。
前提条件	无
参数	• adrAckLimitNew——新值（uint8_t）
返回	无
示例	LORAWAN_SetAdrAckLimit(5);

4.3.2.37 LORAWAN_GetAdrAckLimit

名称	LORAWAN_GetAdrAckLimit
原型	uint8_t LORAWAN_GetAdrAckLimit (void)
摘要	返回自适应数据速率确认限制（ADR_ACK_LIMIT）的值。
说明	每次上行链路帧计数器递增（对于每个新的上行链路，重复传输不会增加计数器的值）时，器件都会使ADR_ACK_CNT计数器递增。 在没有任何下行链路响应的ADR_ACK_LIMIT上行链路之后（ADR_ACK_CNT > = ADR_ACK_LIMIT），它会将ADR确认请求位（ADRACKReq）置1。 网络需要在接下来的ADR_ACK_DELAY帧内用下行链路帧进行响应，在上行链路帧之后接收的任何下行链路帧都将复位ADR_ACK_CNT计数器。
前提条件	无
参数	无
返回	限制的值（uint8_t）。
示例	uint8_t adrAckLimitUsed; adrAckLimitUsed = LORAWAN_GetAdrAckLimit();

4.3.2.38 LORAWAN_SetAdrAckDelay

名称	LORAWAN_SetAdrAckDelay
原型	void LORAWAN_SetAdrAckDelay(uint8_t adrAckDelayNew)
摘要	设置自适应数据速率确认延时（ADR_ACK_DELAY）的值。
说明	每次上行链路帧计数器递增（对于每个新的上行链路，重复传输不会增加计数器的值）时，器件都会使ADR_ACK_CNT计数器递增。 在没有任何下行链路响应的ADR_ACK_LIMIT上行链路之后（ADR_ACK_CNT > = ADR_ACK_LIMIT），它会将ADR确认请求位（ADRACKReq）置1。 网络需要在接下来的ADR_ACK_DELAY帧内用下行链路帧进行响应，在上行链路帧之后接收的任何下行链路帧都将复位ADR_ACK_CNT计数器。
前提条件	无
参数	• adrAckDelayNew——新值（uint8_t）
返回	无
示例	LORAWAN_SetAdrAckDelay(20);

4.3.2.39 LORAWAN_GetAdrAckDelay

名称	LORAWAN_GetAdrAckDelay
原型	uint8_t LORAWAN_GetAdrAckDelay (void)
摘要	返回自适应数据速率确认延时（ADR_ACK_DELAY）的值。
说明	每次上行链路帧计数器递增（对于每个新的上行链路，重复传输不会增加计数器的值）时，器件都会使ADR_ACK_CNT计数器递增。 在没有任何下行链路响应的ADR_ACK_LIMIT上行链路之后（ADR_ACK_CNT > = ADR_ACK_LIMIT），它会将ADR确认请求位（ADRACKReq）置1。 网络需要在接下来的ADR_ACK_DELAY帧内用下行链路帧进行响应，在上行链路帧之后接收的任何下行链路帧都将复位ADR_ACK_CNT计数器。
前提条件	无
参数	无
返回	延时的值（uint8_t）。
示例	uint8_t adrAckDelayUsed; adrAckDelayUsed = LORAWAN_GetAdrAckDelay();

© 2017 Microchip Technology Inc.

4.3.2.40 LORAWAN_SetAckTimeout

名称	LORAWAN_SetAckTimeout
原型	void LORAWAN_SetAckTimeout(uint16_t ackTimeoutNew)
摘要	设置确认超时（ACK_TIMEOUT）的值。
说明	在上行链路传输后，如果终端设备在两个接收窗口之一中的ACK位置1时没有立即接收到帧，则它会重新发送具有相同有效负载和帧计数器的相同帧，持续时间至少为第二个接收窗口后的ACK_TIMEOUT秒。
前提条件	无
参数	• ackTimeoutNew——新的超时值；新值必须以毫秒为单位提供
返回	无
示例	LORAWAN_SetAckTimeout(2500);

4.3.2.41 LORAWAN_GetAckTimeout

名称	LORAWAN_GetAckTimeout
原型	uint16_t LORAWAN_GetAckTimeout (void)
摘要	返回确认超时（ACK_TIMEOUT）的值。
说明	在上行链路传输后，如果终端设备在两个接收窗口之一中的ACK位置1时没有立即接收到帧，则它会重新发送具有相同有效负载和帧计数器的相同帧，持续时间至少为第二个接收窗口后的ACK_TIMEOUT秒。
前提条件	无
参数	无
返回	确认超时的值（uint16_t）
示例	uint16_t ackTimeoutUsed; ackTimeoutUsed = LORAWAN_GetAckTimeout();

4.3.2.42 LORAWAN_SetNumberOfRetransmissions

名称	LORAWAN_SetNumberOfRetransimissions
原型	void LORAWAN_SetNumberOfRetransmissions (uint8_t numberRetransmissions)
摘要	设置重新传输的次数。
说明	如果没有从服务器接收到下行链路确认，该函数用于设置上行链路确认的数据包的重新传输次数。 复位时，重新传输次数默认为7。
前提条件	无
参数	• numberRetransmissions——新值（uint8_t）
返回	无
示例	LORAWAN_SetNumberOfRetransimissions(3);

4.3.2.43 LORAWAN_GetNumberOfRetransmissions

名称	LORAWAN_GetNumberOfRetransimissions
原型	uint8_t LORAWAN_GetNumberOfRetransmissions (void)
摘要	返回重新传输的次数。
说明	如果没有从服务器接收到下行链路确认，该函数用于返回上行链路确认的数据包的重新传输次数。
前提条件	无
参数	无
返回	重新传输次数（uint8_t）
示例	uint8_t noOfRetrUsed; noOfRetrUsed = LORAWAN_GetNumberOfRetransmissions();

4.3.2.44 LORAWAN_SetReceiveWindow2Parameters

名称	LORAWAN_SetReceiveWindow2Parameters
原型	LorawanError_t LORAWAN_SetReceiveWindow2Parameters (uint32_t frequency, uint8_t dataRate)
摘要	设置第二个接收窗口（RX2）的参数。
说明	此函数用于设置第二个接收窗口所使用的数据速率和频率。 接收窗口参数的配置应与服务器配置保持一致。
前提条件	无
参数	• frequency——新频率（必须以Hz为单位提供） • dataRate——新的数据速率
返回	返回LoRaWAN错误类型（LorawanError_t）
示例	LORAWAN_SetReceiveWindow2Parameters(868100000, 3);

© 2017 Microchip Technology Inc.

4.3.2.45 LORAWAN_GetReceiveWindow2Parameters

名称	LORAWAN_GetReceiveWindow2Parameters
原型	void LORAWAN_GetReceiveWindow2Parameters (uint32_t* frequency, uint8_t* dataRate)
摘要	获取第二个接收窗口（RX2）的参数。
说明	此函数用于获取第二个接收窗口所使用的数据速率和频率。 接收窗口参数的配置应与服务器配置保持一致。
前提条件	无
参数	• frequency——指向以Hz为单位的频率的指针（32位值） • data rate——指向数据速率的指针（8位值）
返回	无
示例	uint32_t win2FreqUsed; uint8_t win2DataRateUsed; LORAWAN_GetReceiveWindow2Parameters(&win2FreqUsed, &win2DataRateUsed);

4.3.2.46 LORAWAN_SetBattery

名称	LORAWAN_SetBattery
原型	void LORAWAN_SetBattery (uint8_t batteryLevelNew)
摘要	设置电池。
说明	此函数用于设置与LoRaWAN协议一起使用的设备状态应答帧所需的电池电量。 其值是一个从0到255的十进制数，用于表示电池电量；0表示外部电源，1表示最低电量，254表示最高电量，255表示终端设备无法测量电池电量。
前提条件	无
参数	• batteryLevelNew——新电量值
返回	无
示例	LORAWAN_SetBattery(0);

4.3.2.47 LORAWAN_GetPrescaler

名称	LORAWAN_GetPrescaler
原型	uint16_t LORAWAN_GetPrescaler (void)
摘要	返回占空比预分频比值。
说明	此函数用于返回占空比预分频比。预分频比值只能由服务器通过使用占空比请求帧来配置。 从服务器接收到此命令后，所有使能通道的占空比预分频比都会更改。
前提条件	无
参数	无
返回	预分频比的值（uint16_t）
示例	uint16_t prescalerUsed; prescalerUsed = LORAWAN_GetPrescaler();

4.3.2.48 LORAWAN_SetAutomaticReply

名称	LORAWAN_SetAutomaticReply
原型	void LORAWAN_SetAutomaticReply (bool status)
摘要	设置自动回复模式的状态。
说明	此函数用于设置自动回复的状态。通过使能自动回复，模块将在接收到确认的下行链路消息之后或者在服务器设置帧待处理位后立即发送不含有效负载的数据包。如果设置为OFF，将不会发送自动回复。
前提条件	无
参数	• status——开/关（真/假）
返回	无
示例	LORAWAN_SetAutomaticReply(false);

© 2017 Microchip Technology Inc.

4.3.2.49 LORAWAN_GetAutomaticReply

名称	LORAWAN_GetAutomaticReply
原型	bool LORAWAN_GetAutomaticReply (void)
摘要	返回自动回复模式的状态。
说明	此函数用于返回自动回复的状态。通过使能自动回复，模块将在接收到确认的下行链路消息之后或者在服务器设置帧待处理位后立即发送不含有效负载的数据包。如果设置为OFF，将不会发送自动回复。
前提条件	无
参数	无
返回	返回自动回复的模式（真/假）
示例	bool autoReplyStatus; autoReplyStatus = LORAWAN_GetAutomaticReply();

4.3.2.50 LORAWAN_GetStatus

名称	LORAWAN_GetStatus
原型	uint32_t LORAWAN_GetStatus (void)
摘要	返回模块的状态。
说明	此函数将返回模块的当前状态。返回的值是以十六进制形式表示的位掩码。关于位掩码的意义，请参见 *“RN2903 LoRa™ Technology Module Command Reference User’s Guide”*（DS40001811）。
前提条件	无
参数	无
返回	模块的状态（uint32_t）
示例	uint32_t moduleStatus; moduleStatus = LORAWAN_GetStatus();

4.3.2.51 LORAWAN_GetLinkCheckMargin

名称	LORAWAN_GetLinkCheckMargin
原型	uint8_t LORAWAN_GetLinkCheckMargin (void)
摘要	返回表示解调余量的十进制数。
说明	此函数将返回在最后一个连接检查应答帧中接收到的解调余量。有关值的说明，请参见LoRaWAN规范1.0文档。
前提条件	无
参数	无
返回	余量值（uint8_t）
示例	uint8_t linkCheckMarginUsed; linkCheckMarginUsed = LORAWAN_GetLinkCheckMargin();

4.3.2.52 LORAWAN_GetLinkCheckGwCnt

名称	LORAWAN_GetLinkCheckGwCnt
原型	uint8_t LORAWAN_GetLinkCheckGwCnt (void)
摘要	返回表示网关数的十进制数。
说明	此函数将返回在最后一个连接检查应答中成功接收到最后一个连接检查请求帧命令的网关的数量。
前提条件	无
参数	无
返回	网关数量（uint8_t）
示例	uint8_t linkCheckGwCntUsed; linkCheckGwCntUsed = LORAWAN_GetLinkCheckGwCnT();

© 2017 Microchip Technology Inc.

4.3.2.53 LORAWAN_GetFrequency

名称	LORAWAN_GetFrequency
原型	uint32_t LORAWAN_GetFrequency (uint8_t channelId)
摘要	返回给定通道的频率。
说明	此函数用于返回以十进制形式输入的所请求“channelId”上的频率。
前提条件	无
参数	• channelId——请求的通道（uint8_t）
返回	给定通道的频率（返回的值以Hz为单位——uint32_t）
示例	uint32_t freqUsed; freqUsed = LORAWAN_GetFrequency(4);

4.3.2.54 LORAWAN_SetDataRange

名称	LORAWAN_SetDataRange
原型	LorawanError_t LORAWAN_SetDataRange (uint8_t channelId, uint8_t dataRangeNew)
摘要	为给定通道设置新数据速率范围。
说明	此函数用于为给定“channelId”设置工作数据速率范围（最小值到最大值）。之后，模块即可以在最小值和最大值之间改变数据速率。 有关数据速率的实际值和相应的扩频因子（Spreading Factor，SF）的信息，请参见LoRaWAN规范V1.0文档。
前提条件	无
参数	• channelId——请求的通道 • dataRangeNew——前四个MSB表示最大值，后四个LSB包含最小值。
返回	无
示例	设置通道13的数据速率范围（1到3）： MacSetDataRange(13, 0x31); // (0x31 -> 0011 0001)

4.3.2.55 LORAWAN_GetDataRange

名称	LORAWAN_GetDataRange
原型	uint8_t LORAWAN_GetDataRange (uint8_t channelId)
摘要	返回给定通道的数据速率范围。
说明	此函数用于返回给定“channelId”的工作数据速率范围（最小值到最大值）。
前提条件	无
参数	• channelId——请求的通道
返回	返回最小和最大数据速率范围（uint8_t）。前四个MSB表示最大值，后四个LSB包含最小值。
示例	uint8_t dataRangeUsed; dataRangeUsed = LORAWAN_GetDataRange(3);

4.3.2.56 LORAWAN_SetChannelIdStatus

名称	LORAWAN_SetChannelIdStatus
原型	LorawanError_t LORAWAN_SetChannelIdStatus (uint8_t channelId, bool statusNew)
摘要	为给定通道设置新状态。
说明	此函数用于设置给定“channelId”的工作状态。
前提条件	无
参数	• channelId——表示通道编号的十进制数 • statusNew——表示开/关（真/假）状态的值
返回	返回LoRaWAN错误类型（LorawanError_t）
示例	LORAWAN_SetChannelIdStatus(5, DISABLED);

4.3.2.57 LORAWAN_GetChannelIdStatus

名称	LORAWAN_GetChannelIdStatus
原型	bool LORAWAN_GetChannelIdStatus (uint8_t channelId)
摘要	返回给定通道的状态。
说明	此函数用于返回给定“channelId”的状态。
前提条件	无
参数	• channelId——表示通道编号的十进制数
返回	通道状态——使能/禁止（真/假）
示例	bool channel3Status; channel3Status = LORAWAN_GetChannelIdStatus(3);

© 2017 Microchip Technology Inc.

4.3.2.58 LORAWAN_Pause

名称	LORAWAN_Pause
原型	uint32_t LORAWAN_Pause (void)
摘要	暂停LoRaWAN协议栈。
说明	此函数用于暂停LoRaWAN协议栈功能以允许收发器（无线电）配置。通过使用“mac pause”，可以在LoRaWAN协议上行链路应用程序和LoRaWAN协议接收窗口之间生成无线电命令。 此函数将在一定时间间隔（以毫秒为单位）内进行回复，以告知此时使用收发器不会影响LoRaWAN功能。
前提条件	无
参数	无
返回	返回以毫秒为单位的数字，表示在不影响功能的情况下可以暂停多长时间。不能暂停时返回“0”，处于空闲模式时返回最大值。
示例	uint32_t timeToPauseInMs; timeToPauseInMs = LORAWAN_Pause();

4.3.2.59 LORAWAN_Resume

名称	LORAWAN_Resume
原型	void LORAWAN_Resume (void)
摘要	恢复LoRaWAN协议栈功能。
说明	此函数用于恢复LoRaWAN协议栈功能，以便在暂停后继续正常工作。
前提条件	无
参数	无
返回	无
示例	LORAWAN_Resume();

4.3.2.60 LORAWAN_LinkCheckConfigure

名称	LORAWAN_LinkCheckConfigure
原型	void LORAWAN_LinkCheckConfigure (uint16_t period)
摘要	设置连接检查过程的时间间隔。
说明	此函数用于设置定期触发连接检查过程的时间间隔。<值>为0时将禁止连接检查过程。 当时间间隔到期时，将被发送到服务器的下一个应用程序数据包将包括连接检查MAC命令。 有关连接检查配置的更多信息，请参见LoRaWAN规范V1.0文档。
前提条件	无
参数	• period——新周期值；必须以秒为单位提供
返回	无
示例	LORAWAN_LinkCheckConfigure(10);

4.3.2.61 LORAWAN_ForceEnable

名称	LORAWAN_ForceEnable
原型	void LORAWAN_ForceEnable (void)
摘要	禁止立即静默状态。
说明	网络可以发出特定命令以要求终端设备立即静默。该机制用于禁止模块的任何进一步通信，有效地将其与网络隔离。 如果使用此功能，则在收到此网络命令后，将恢复模块的连接，允许其发送数据。
前提条件	无
参数	无
返回	无
示例	LORAWAN_ForceEnable();

4.3.2.62 LORAWAN_Reset（用于EU433/EU868 ISM频段）

名称	LORAWAN_Reset
原型	void LORAWAN_Reset (IsmBand_t ismBandNew)
摘要	此函数将自动复位软件LoRaWAN协议栈，并使用所选ISM频段的参数对其进行初始化。
说明	此API将为大多数LoRaWAN参数设置默认值。在此命令之前设置的所有内容将丢失其设置值，重新初始化为默认值，包括将加密密钥设置为0。
前提条件	无
参数	• ismBandNew——新频段（IsmBand_t）。
返回	无
示例	LORAWAN_Reset(ISM_EU868);

© 2017 Microchip Technology Inc.

4.3.2.63 LORAWAN_Reset（用于NA915 ISM频段）

名称	LORAWAN_Reset
原型	void LORAWAN_Reset (void)
摘要	此函数将自动复位软件LoRaWAN协议栈，并使用所选ISM频段的参数对其进行初始化。
说明	此API将为大多数LoRaWAN参数设置默认值。在此命令之前设置的所有内容将丢失其设置值，重新初始化为默认值，包括将加密密钥设置为0。
前提条件	无
参数	无
返回	无
示例	LORAWAN_Reset();

4.3.2.64 LORAWAN_SetFrequency（仅用于EU433/EU868）

名称	LORAWAN_SetFrequency
原型	LorawanError_t LORAWAN_SetFrequency (uint8_t channelId, uint32_t frequencyNew)
摘要	此函数用于将所请求“channelId”上的频率设置为新值。
说明	此API将为大多数LoRaWAN参数设置默认值。在此命令之前设置的所有内容将丢失其设置值，重新初始化为默认值，包括将加密密钥设置为0。
前提条件	无
参数	• channelId——给定的通道 • frequencyNew——新频率值（值必须以Hz为单位提供）。
返回	返回LoRaWAN错误类型（LorawanError_t）

示例	uint8_t channel4 = 4;
	uint32_t freqCh4 = 868500000;
	
	LORAWAN_SetFrequency(channel4, freqCh4);

4.3.2.65 LORAWAN_SetDutyCycle（仅用于EU433/EU868）

名称	LORAWAN_SetDutyCycle
原型	LorawanError_t LORAWAN_SetDutyCycle (uint8_t channelId, uint16_t dutyCycleValue)
摘要	此函数用于设置占空比。
说明	此函数用于设置给定通道上的占空比。
前提条件	无
参数	• channelId——给定的通道 • dutyCycleValue——占空比的值； dutyCycleValue = (100 / X) - 1，其中X是以百分比表示的占空比。 更多详细信息，请参见LoRaWAN规范V0.1文档。
返回	返回LoRaWAN错误类型（LorawanError_t）

示例	uint8_t channel5 = 5;
	uint16_t dutyCycle5 = 9;
	
	LORAWAN_SetDutyCycle(channel5, dutyCycle5);

4.3.2.66 LORAWAN_GetDutyCycle（仅用于EU433/EU868）

名称	LORAWAN_GetDutyCycle
原型	uint16_t LORAWAN_GetDutyCycle (uint8_t channelId)
摘要	此函数用于返回给定通道的占空比。
说明	此函数用于返回给定通道的占空比的值。 返回值使用以下公式计算： dutyCycleValue = (100 / X) - 1，其中X是以百分比表示的占空比。 更多详细信息，请参见LoRaWAN规范V1.0文档。
前提条件	无
参数	• channelId——给定的通道。
返回	返回所请求通道占空比（uint16_t）

示例	uint8_t channel6 = 6;
	uint16_t dutyCycleUsed;
	
	dutyCycleUsed = LORAWAN_GetDutyCycle(channel6);

© 2017 Microchip Technology Inc.

4.3.2.67 LORAWAN_GetISMBand（仅用于EU433/EU868）

名称	LORAWAN_GetISMBand
原型	uint8_t LORAWAN_GetIsmBand(void)
摘要	返回配置的ISM频段。
说明	此函数用于返回配置的ISM频段。
前提条件	无
参数	无
返回	返回ISM频段类型（ISMBand_t）
示例	uint8_t ISMBandUsed; ISMBandUsed = LORAWAN_GetISMBand();

4.3.2.68 LORAWAN_Mainloop

名称	LORAWAN_Mainloop
原型	void LORAWAN_Mainloop (void)
摘要	LoRaWAN主循环函数
说明	此函数用于运行系统定时器和检查DIO引脚。它必须在<main>函数内部的while(1)循环中调用（每次循环调用一次）。
前提条件	无
参数	无
返回	无

示例	while(1)
	{
	  LORAWAN_Mainloop();
	}

4.3.2.69 LORAWAN_SetClass

名称	LORAWAN_SetClass
原型	void LORAWAN_SetClass (LoRaClass_t deviceClass)
摘要	设置LoRaWAN设备类别
说明	此函数用于将LoRaWAN协议栈类别设置为A类或C类。
前提条件	无
参数	• deviceClass——新类别（LoRaClass_t）
返回	无
示例	LORAWAN_SetClass(CLASS_C)

4.3.2.70 LORAWAN_GetClass

名称	LORAWAN_GetClass
原型	LoRaClass_t LORAWAN_GetClass (void);
摘要	返回LoRaWAN设备类别
说明	此函数用于返回LoRaWAN协议栈类别。
前提条件	无
参数	无
返回	返回LoRaWANClass类型（LoRaClass_t）
示例	LoRaClass_t deviceClass; deviceClass = LOWRAWAN_GetClass();

4.3.2.71 LORAWAN_SetMcast

名称	LORAWAN_SetMcast
原型	LorawanError_t LORAWAN_SetMcast(bool status);
摘要	设置多播的状态
说明	此函数用于使能或禁止多播操作。
前提条件	当使能多播时，必须设置其参数（mcastNetworkSessionKey、mcastApplicationSessionKey和mcastDeviceAddressNew）且必须加入网络。
参数	• status——真或假（bool）
返回	返回LoRaWAN错误类型（LorawanError_t）
示例	LORAWAN_SetMcast (true)

4.3.2.72 LORAWAN_GetMcast

名称	LORAWAN_GetMcast
原型	bool LORAWAN_GetMcast(void);
摘要	返回多播的状态。
说明	此函数用于返回多播的状态。
前提条件	无
参数	无
返回	status——真或假（bool）
示例	bool status; status = LORAWAN_GetMcast();

© 2017 Microchip Technology Inc.

4.3.2.73 LORAWAN_SetMcastApplicationSessionKey

名称	LORAWAN_SetMcastApplicationSessionKey
原型	void LORAWAN_SetMcastApplicationSessionKey (uint8_t *mcastApplicationSessionKeyNew);
摘要	设置多播应用程序会话密钥。
说明	此函数用于设置多播应用程序会话密钥（McastAppSKey）。McastAppSKey是特定于一组终端设备的应用程序会话密钥，用于对应用程序特定的多播数据消息的有效负载字段进行加密/解密，还用于计算/验证有效负载中可能包含的应用程序级MIC（消息完整性代码）。McastAppSKey是一个16字节的数组。
前提条件	指针必须由调用程序分配
参数	• mcastApplicationSessionKeyNew——存储值的缓冲区。
返回	无
示例	uint8_t mcastAppSKey[16] = {0x3C, 0x8F, 0x26, 0x27, 0x39, 0xBF, 0xE3, 0xB7, 0xBC, 0x08, 0x26, 0x99, 0x1A, 0xD0,0x50, 0x4D}; LORAWAN_SetApplicationSessionKey(mcastAppSKey);

4.3.2.74 LORAWAN_GetMcastApplicationSessionKey

名称	LORAWAN_GetMcastApplicationSessionKey
原型	void LORAWAN_GetMcastApplicationSessionKey (uint8_t *mcastApplicationSessionKeyNew);
摘要	获取多播应用程序会话密钥。
说明	此函数用于获取多播应用程序会话密钥（McastAppSKey）。McastAppSKey是特定于一组终端设备的应用程序会话密钥，用于对应用程序特定的多播数据消息的有效负载字段进行加密/解密，还用于计算/验证有效负载中可能包含的应用程序级MIC（消息完整性代码）。McastAppSKey是一个16字节的数组。
前提条件	指针必须由调用程序分配
参数	• mcastApplicationSessionKey——将存储值的缓冲区。
返回	无
示例	uint8_t mcastAppSKey[16]; LORAWAN_GetApplicationSessionKey(mcastAppSKey);

4.3.2.75 LORAWAN_SetMcastNetworkSessionKey

名称	LORAWAN_SetMcastNetworkSessionKey
原型	void LORAWAN_SetMcastNetworkSessionKey (uint8_t *mcastNetworkSessionKeyNew);
摘要	设置多播网络会话密钥。
说明	此函数用于设置多播网络会话密钥（McastNwkSKey）。
前提条件	指针必须由调用程序分配
参数	• mcastNetworkSessionKeyNew——存储值的缓冲区。
返回	无
示例	uint8_t mcastNwkSKey[16] = {0x2B, 0x7E, 0x15, 0x16, 0x28, 0xAE, 0xD2, 0xA6, 0xAB, 0xF7, 0x15, 0x88, 0x09, 0xCF, 0x4F, 0x3C}; LORAWAN_SetMcastNetworkSessionKey(mcastNwkSKey);

4.3.2.76 LORAWAN_GetMcastNetworkSessionKey

名称	LORAWAN_GetMcastNetworkSessionKey
原型	void LORAWAN_GetMcastNetworkSessionKey (uint8_t *mcastNetworkSessionKeyNew);
摘要	获取多播网络会话密钥。
说明	此函数用于获取多播网络会话密钥（McastNwkSKey）。
前提条件	指针必须由调用程序分配
参数	• mcastNetworkSessionKey——将存储值的缓冲区。
返回	无
示例	uint8_t mcastNwkSKey[16]; LORAWAN_GetMcastNetworkSessionKey(mcastNwkSKey);

© 2017 Microchip Technology Inc.

4.3.2.77 LORAWAN_SetMcastDeviceAddress

名称	LORAWAN_SetMcastDeviceAddress
原型	void LORAWAN_SetMcastDeviceAddress (uint32_t mcastDeviceAddressNew);
摘要	设置一组终端设备的多播地址。
说明	此函数用于设置一组终端设备的多播地址组（McastDevAddr）。McastDevAddr是当前网络中的一组终端设备的32位标识符。
前提条件	无
参数	• mcastDeviceAddressNew——要设置的新地址的值。
返回	无
示例	uint32_t mcastDevAddr = 0x11223344; LORAWAN_SetMcastDeviceAddress (mcastDevAddr);

4.3.2.78 LORAWAN_GetMcastDeviceAddress

名称	LORAWAN_GetMcastDeviceAddress
原型	uint32_t LORAWAN_GetMcastDeviceAddress (void);
摘要	返回一组终端设备的多播地址。
说明	此函数用于获取一组终端设备的多播地址（McastDevAddr）。McastDevAddr是当前网络中的一组终端设备的32位标识符。
前提条件	无
参数	无
返回	多播设备地址（uint32_t值）
示例	uint32_t mcastDevAddr; mcastDevAddr = LORAWAN_GetMcastDeviceAddress();

4.3.2.79 LORAWAN_SetMcastDownCounter

名称	LORAWAN_SetMcastDownCounter
原型	void LORAWAN_SetMcastDownCounter(uint32_t newCnt);
摘要	设置多播下行链路计数器。
说明	此函数用于设置多播下行链路计数器。它可用于加入已存在的多播组的设备。
前提条件	无
参数	• newCnt——新的下行链路计数器值（uint32_t）
返回	无
示例	uint32_t newCnt = 12345; LORAWAN_SetMcastDownCounter(newCnt);

4.3.2.80 LORAWAN_GetMcastDownCounter

名称	LORAWAN_GetMcastDownCounter
原型	uint32_t LORAWAN_GetMcastDownCounter();
摘要	返回多播下行链路计数器。
说明	此函数用于返回多播下行链路计数器。
前提条件	无
参数	无
返回	多播下行链路计数器（uint32_t）
示例	uint32_t newCnt; newCnt = LORAWAN_SetMcastDownCounter();

4.3.2.81 LORAWAN_GetState

名称	LORAWAN_GetState
原型	uint8_t LORAWAN_GetState(void)
摘要	返回LoRaWAN协议栈的状态。
说明	此函数用于返回LoRaWAN协议栈的状态。 可能的LoRaWAN A类/C类状态如下： • IDLE • TRANSMISSION_OCCURRING • BEFORE_RX1 • RX1_OPEN • BETWEEN_RX1_RX2 • RX2_OPEN • RETRANSMISSION_DELAY • ABP_DELAY • CLASS_C_RX2_1_OPEN • CLASS_C_RX2_2_OPEN
前提条件	无
参数	无
返回	返回LoRaWAN协议栈的状态（uint8_t）
示例	uint8_t stateOfLorawan; stateOfLorawan = LORAWAN_GetState(void);

© 2017 Microchip Technology Inc.

第5章 构建基于LoRaWAN™的应用程序

5.1 概述

第3.3节“RN2XX3模块的LoraWAN™协议栈生成”中生成的LoRaWAN协议栈文件可用于构建基于LoRaWAN的自定义应用程序。

属于自定义应用程序的所有代码必须放在LoRaWAN协议栈文件之外（例如，代码可以放在`main.c`或任何其他非LoRaWAN相关文件中）。

> **注：** 不得修改LoRaWAN生成的文件中的代码，否则可能导致LoRaWAN协议栈无法正常工作。

本章介绍如何构建一个基于LoRaWAN的简单应用程序，借此不断向服务器发送文本消息（字符串）。终端设备必须进行初始化、激活配置和激活（必须加入网络）后才能发送数据。这四个状态将在以下几节中介绍。

5.2 基本LoRaWAN A类协议栈操作

5.2.1 概述

对于基本LoRaWAN A类协议栈操作，可标识以下四个状态：

1. 初始化
2. 激活配置
3. 激活
4. 通信

基本操作将在以下几节中介绍。

5.2.2 初始化

在初始化操作中，必须初始化协议栈。这通过调用`LORAWAN_Init`函数来完成。在允许外设和全局中断之后，应立即在主函数内的`while(1)`无限循环之前调用此函数。

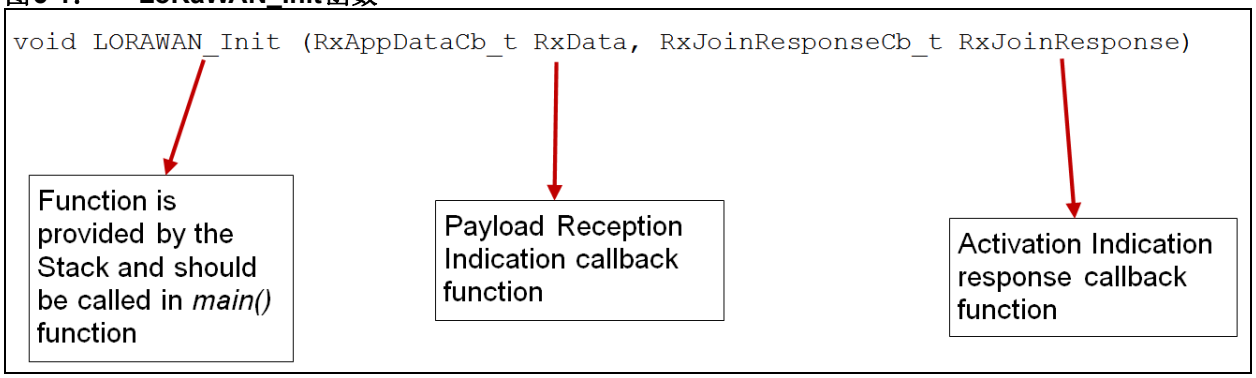
有关如何使用此函数的更多详细信息，请参见第5.6节“基于LoRaWAN的自定义应用程序示例”。

`LORAWAN_Init`函数用于初始化LoRaWAN协议栈和无线电收发器。必须定义两个回调函数，并将其用作`LORAWAN_Init`函数的参数。

必须定义的两个回调函数用于以下目的：

- 有效负载接收指示
- 激活指示响应

图5-1： LoRaWAN_Init函数



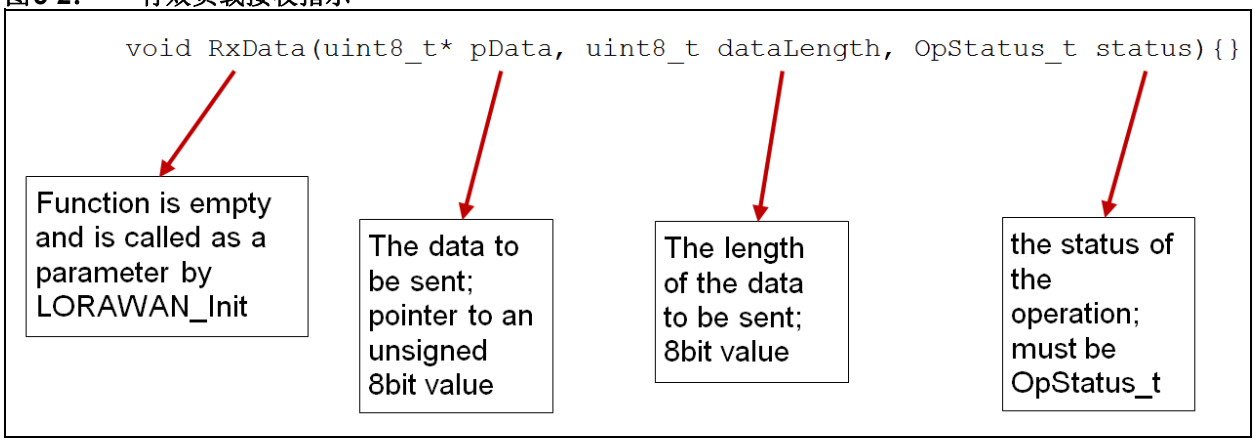
有关LORAWAN_Join使用的回调函数的详细信息，请参见第5.2.2.1节“有效负载接收指示（有效负载接收回调）”和第5.2.2.2节“激活指示响应”。

5.2.2.1 有效负载接收指示（有效负载接收回调）

有效负载接收回调可以是一个空函数，其参数如下：

- uint8_t* pData——指向8位数据的指针；这是要发送的实际数据
- uint8_t dataLength——要发送的数据的长度
- OpStatus_t status——操作状态。这实际上是指示是否已接收的参数。状态变量的返回值可以是：
  - MAC_NOT_OK——LoRaWAN操作失败
  - MAC_OK——LoRaWAN操作成功
  - RADIO_NOT_OK——无线电操作失败
  - RADIO_OK——无线电操作成功
  - INVALID_BUFFER_LENGTH——已尝试重新传输，缓冲区太大，这是因为扩频因子发生变化。

图5-2： 有效负载接收指示



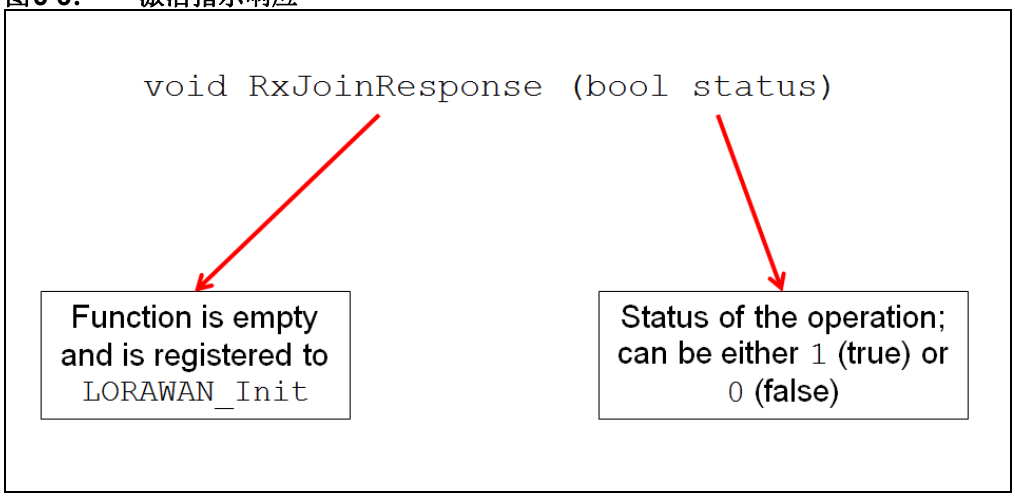
© 2017 Microchip Technology Inc.

5.2.2.2 激活指示响应

激活指示回调函数可以是一个空函数，但必须具有以下参数：

- bool状态——指示来自网络的激活响应的二进制值：
  - status = 1（真）——设备已连接到网络
  - status = 0（假）——设备未连接到网络

图5-3： 激活指示响应



5.2.3 激活配置

在激活配置阶段中，服务器密钥必须传送到LoRaWAN协议栈。激活可以采用以下两种方式：

- 个性化激活（Activation-By-Personalization，ABP）
- 无线激活（Over-the-Air Activation，OTAA）

5.2.3.1 个性化激活配置

为了使用个性化激活过程将终端设备连接到服务器，需要借助三个API将两个密钥和一个惟一标识符传送到LoRaWAN协议栈。两个密钥和惟一标识符如下：

- 网络会话密钥——NwkSKey是特定于终端设备的网络会话密钥。它供网络服务器和终端设备用于计算和验证所有数据消息的MIC（消息完整性代码），以确保数据完整性。它还用于加密和解密网络管理消息的有效负载字段。
- 应用程序会话密钥——AppSKey是特定于终端设备的应用程序会话密钥。它供应用程序服务器和终端设备用于加密和解密应用特定数据消息的有效负载字段。
- 设备地址——由32位组成的DevAddr，用于标识当前网络中的终端设备。其格式如图5-4所示。

图5-4： 设备地址

Bit#	[31..25]	[24..0]
DevAddr bits	NwkID	NwkAddr

高7位用作网络标识符（NwkID）以区分不同网络运营商的区域重叠网络的地址并解决漫游问题。低25位是终端设备的网络地址（NwkAddr），可以由网络管理员任意分配。

网络会话密钥示例

NWSKEY：`0x2B7E151628AED2A6ABF7158809CF4F3C`

网络会话密钥由用户选择，并且在终端设备和服务器上必须相同。为了向终端设备提供此信息，必须将NWSKEY拆分为多个字节并定义为16个值的数组。

图5-5：　网络会话密钥拆分示例

2B	7E	15	16	28	AE	D2	A6	AB	F7	15	88	09	CF	4F	3C
byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7	byte 8	byte 9	byte 10	byte 11	byte 12	byte 13	byte 14	byte 15	byte 16

可以通过调用LORAWAN_SetNetworkSessionKey API将NWKSKEY传送到LoRaWAN协议栈：

```
LORAWAN_SetNetworkSessionKey(nwkSKey);
```

> **注：**　参数nwkSKey必须预先定义为16个元素（8位值）的数组。

应用程序会话密钥示例

APPSKEY：`0x3C8F262739BFE3B7BC0826991AD0504D`

应用程序会话密钥由用户选择，并且在终端设备和服务器上必须相同。为了向终端设备提供此信息，必须将APPSKEY拆分为多个字节并定义为16个值的数组。

图5-6：　应用程序会话密钥拆分示例

3C	8F	26	27	39	BF	E3	B7	BC	08	26	99	1A	D0	50	4D
byte 1	byte 2	byte 3	byte 4	byte 5	byte 6	byte 7	byte 8	byte 9	byte 10	byte 11	byte 12	byte 13	byte 14	byte 15	byte 16

可以通过调用LORAWAN_SetApplicationSessionKey API将APPSKEY传送到LoRaWAN协议栈：

```
LORAWAN_SetApplicationSessionKey(appSKey);
```

> **注：**　参数AppSKey必须预先定义为16个元素（8位值）的数组。

设备地址示例：DEVADDR：`0x11223344`

设备地址由用户选择，并且在终端设备和服务器上必须相同。可以通过调用LORAWAN_SetDeviceAddress API将DEVADDR传送到LoRaWAN协议栈：

```
LORAWAN_SetDeviceAddress(devAddr);
```

> **注：**　参数devAddr必须预先定义为32位值变量或常量。

© 2017 Microchip Technology Inc.

5.2.3.2 无线激活配置

为了使用无线激活过程将终端设备连接到服务器，需要借助三个API将三条信息传送到LoRaWAN协议栈。

- 应用程序标识符（AppEUI）——AppEUI是惟一标识终端设备应用程序提供方（即所有者）的IEEE EUI64地址空间中的全局应用程序ID。AppEUI在激活过程执行之前即存储在终端设备中。
- 终端设备标识符（DevEui）——DevEUI是惟一标识终端设备的IEEE EUI64地址空间中的全局终端设备ID。
- 应用程序密钥（AppKey）——AppKey是特定于终端设备的AES-128应用程序密钥（由应用程序所有者分配给终端设备），并且很可能来源于应用程序特定的根密钥（仅对应用程序提供商已知且由其控制）。每当终端设备通过无线激活加入网络时，便会使用AppKey获取特定于该终端设备的会话密钥NwkSKey和AppSKey，从而加密和验证网络通信和应用数据。

可以通过使用LORAWAN_SetApplicationEui API将AppEUI传送到LoRaWAN协议栈：

```
LORAWAN_SetApplicationEui(applicationEuiNew);
```

> **注：** 参数applicationEuiNew必须预先定义为8字节数组。

可以通过使用LORAWAN_SetDeviceEui API将DevEUI传送到LoRaWAN协议栈：

```
LORAWAN_SetDeviceEui (deviceEuiNew);
```

> **注：** 参数deviceEuiNew必须预先定义为8字节数组。

可以通过使用LORAWAN_SetApplicationKey API将AppKey传送到LoRaWAN协议栈：

```
LORAWAN_SetApplicationKey (applicationKeyNew);
```

> **注：** 参数applicationKeyNew必须预先定义为16字节数组。

5.2.4 激活

在激活阶段，终端设备以ABP模式或OTAA模式向服务器发送加入请求。

5.2.4.1 个性化激活

在某些情况下，可以通过个性化方式来激活终端设备。个性化激活将绕过“加入请求 - 加入接受”过程，直接将终端设备连接到特定网络。

通过个性化方式激活终端设备意味着将DevAddr和两个会话密钥NwkSKey和AppSKey（而不是DevEUI、AppEUI和AppKey）直接存储到终端设备中。终端设备配有启动时参与特定LoRa网络所需的信息。

每个设备应该有一组惟一的NwkSKey和AppSKey。对一个设备的密钥进行折衷处理时，应以不影响其他设备的通信安全为前提。创建这些密钥时，应确保其无法借助公开信息（例如节点地址）推断出来。

为了在个性化激活模式下将终端设备连接到网络，必须使用以下API和参数：

```
LORAWAN_Join(ABP);
```

必须在主函数内的`while (1)`循环之前调用此API。有关基于LoRaWAN协议栈构建软件的更多详细信息，请参见**第5.6节“基于LoRaWAN的自定义应用程序示例”**。

5.2.4.2 无线激活

对于无线激活，终端设备必须先完成加入过程，然后才能与网络服务器进行数据交换。每次终端设备丢失会话上下文信息时，都必须重新完成加入过程。

加入过程要求终端设备在开始加入过程之前使用以下信息进行个性化：全局惟一终端设备标识符（DevEUI）、应用程序标识符（AppEUI）和AES-128密钥（AppKey）。

为了在无线激活模式下将终端设备连接到网络，必须使用以下API和参数：

```
LORAWAN_Join(OTAA);
```

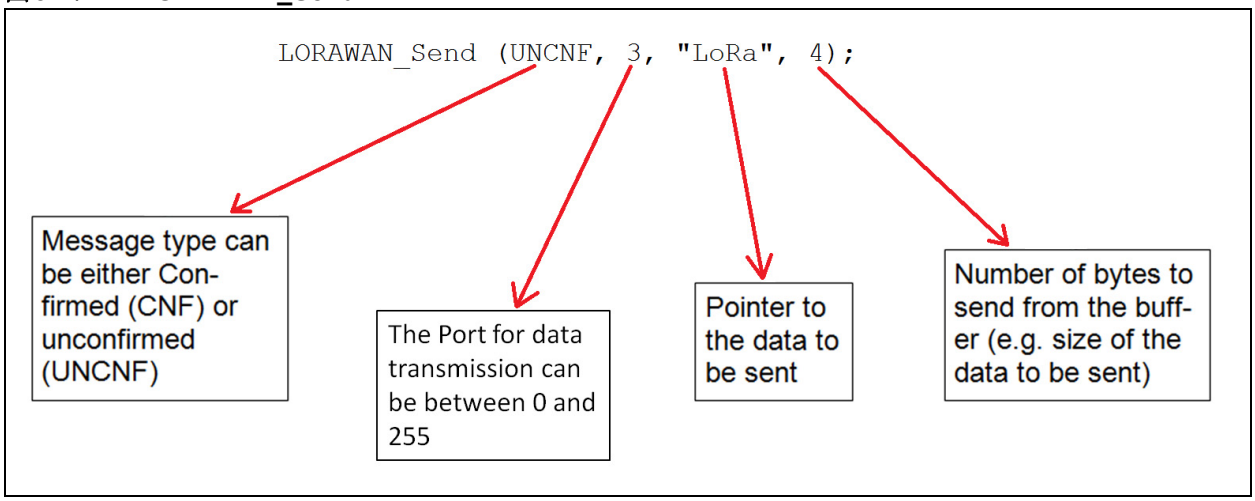
5.2.5 通信

设备连接到网络后，用户的自定义应用程序就能够向服务器发送数据。可以使用LORAWAN_Send API完成数据发送：

```
LORAWAN_Send (UNCNF, 3, “LoRa”, 4);
```


© 2017 Microchip Technology Inc.

图5-7： LORAWAN_Send API



5.3 基本LoRaWAN C类协议栈操作

5.3.1 概述

实现C类选项的终端设备用于不必考虑电流消耗并因此无需尽量缩短接收时间的应用。

C类终端设备应尽可能频繁地打开RX2窗口。根据A类定义，如果终端设备在RX1窗口期间既不发送也不接收，则它将在RX2窗口期间处于监听状态。

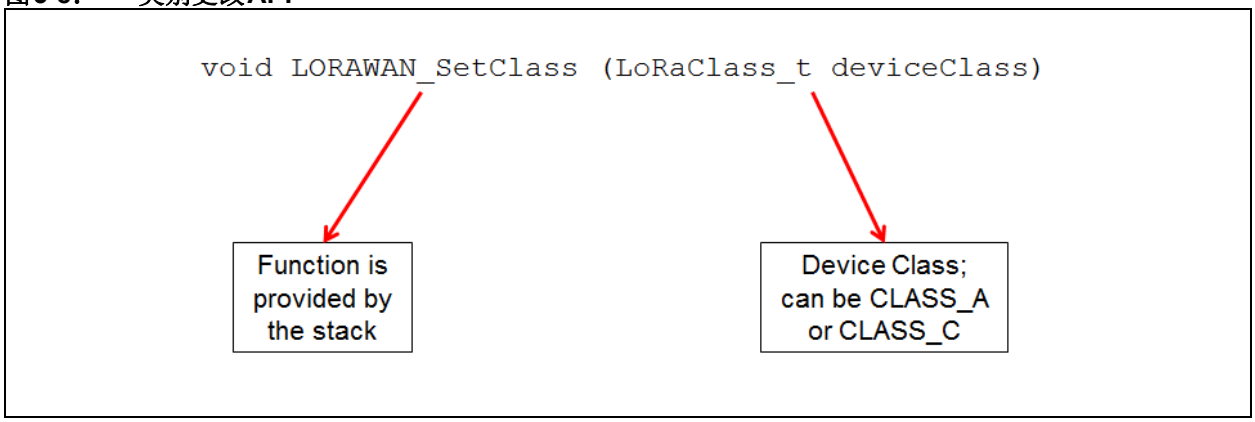
有关C类参数的更多详细信息，请参见LoRaWAN规范文档。

对于以C类启动的设备，其初始化过程与A类相同。有关初始化过程的详细信息，请参见第5.2节“基本LoRAWAN A类协议栈操作”。

5.3.2 类别更改

设备可以作为A类设备或C类设备启动。如果在运行时需要更改操作类别，可以使用LORAWAN_SetClass API来完成。更多详细信息，请参见第4.3节“应用程序编程接口”。

图5-8： 类别更改API



5.3.3 C类连续接收

在设备更改为C类或以C类启动后，如要进入连续接收模式，必须成功完成到服务器的上行链路。有关通信过程的更多信息，请参见第5.2.5节“通信”。

5.3.4 C类多播消息

消息可以是“单播”或“多播”。多播消息发送到多播组内的多个终端设备。多播组的所有设备必须共享相同的多播地址和相关加密密钥。C类设备可以接收多播下行链路帧。多播地址以及相关网络会话密钥和应用程序会话密钥必须来自应用层。MCC LoRaWAN库提供了几个API，支持用户开发基于LoRaWAN的应用程序，该应用程序还包括C类多播选项。更多详细信息，请参见第4.3节“应用程序编程接口”。

5.4 构建基于LORAWAN的自定义应用程序

5.4.1 概述

在前几节中介绍的所有API都必须在main.c文件中按特定顺序调用，该文件由MCC自动生成。

5.4.2 中断允许

必须在main.c文件中允许全局中断和外设中断。为了允许这些中断，必须在main.c文件中为全局中断和外设中断的代码行取消注释，如例5-1所示：

例5-1： MAIN.C文件

```
// Enable the Global Interrupts
   INTERRUPT_GlobalInterruptEnable();

// Enable the Peripheral Interrupts
   INTERRUPT_PeripheralInterruptEnable();
```

5.4.3 LoRaWAN Mainloop函数

LORAWAN_Mainloop()函数正在运行LoRaWAN协议栈，因此必须在while(1)无限循环中调用它。自定义应用程序的所有其他用户自定义函数都必须在while(1)无限循环中的LORAWAN_Mainloop()函数调用后调用。

例5-2： LoRaWAN™ MAINLOOP函数

```
.........................
while (1)
    {
       LORAWAN_Mainloop();
       ..............................................
       ..............................................
    }
.........................
```


© 2017 Microchip Technology Inc.

5.5 LoRaWAN协议栈加密

LoRaWAN协议栈使用AES-128加密。所使用的加密方案基于IEEE 902.15.4/2006附录B [IEEE802154]中描述的通用算法，该算法使用密钥长度为128位的AES。有关LoRaWAN协议栈的加密/解密的更多详细信息，请参见LoRaWAN规范V1.0文档。

Microchip不能分发AES.h和AES.c加密文件。由于许可条款的差异，LoRaWAN协议栈使用的AES引擎需要单独添加。用户自行负责获取加密文件。必须在项目中手动替换这两个文件，因为与协议栈一起生成的文件只包含一个通知用户缺少AES加密的错误。

加密文件可以通过以下链接下载：http://www.microchip.com/Developmenttools/ProductDetails.aspx?PartNO=SW300052。

在下载数据加密库V2.6.zip并提取AES.h和AES.c文件后，用户必须确保文件不包含不受XC编译器支持的任何关键字（例如，“rom”关键字必须删除，因为它不受XC8编译器支持）。

5.6 基于LORAWAN的自定义应用程序示例

LORAWAN_Mainloop()函数正在运行LoRaWAN协议栈，因此必须在while(1)无限循环中调用它。自定义应用程序的所有其他用户自定义函数都必须在while(1)无限循环中的LORAWAN_Mainloop()函数调用后调用。

以下代码示例基于使用LoRaWAN库的MCC生成的代码（见**第3章“运行在RN2XX3模块上的LoRaWAN™库插件”**），可以在RN2XX3模块上运行。

例5-3： MAIN.C文件演示代码

软件许可协议

Microchip Technology Incorporated（“公司”）随附提供的软件旨在提供给您（该公司的客户）使用，仅限于且只能在该公司制造的产品上使用。

该软件为公司和/或其供应商所有，并受适用的版权法保护。版权所有。任何违反前述限制的使用将使其用户遭受适用法律的刑事制裁，并承担违背此许可的条款和条件的民事责任。

该软件“按现状”提供。不提供保证，无论是明示的、暗示的还是法定的保证。这些保证包括（但不限于）对出于某一特定目的应用此软件的适销性和适用性默示的保证。在任何情况下，公司都将不会对任何原因造成的特别的、偶然的或间接的损害负责。

```
#include "mcc_generated_files/mcc.h”

uint8_t nwkSKey[16] = {0x2B, 0x7E, 0x15, 0x16, 0x28, 0xAE, 0xD2, 0xA6, 0xAB,
0xF7, 0x15, 0x88, 0x09, 0xCF, 0x4F, 0x3C};
uint8_t appSKey[16] = {0x3C, 0x8F, 0x26, 0x27, 0x39, 0xBF, 0xE3, 0xB7, 0xBC,
0x08, 0x26, 0x99, 0x1A, 0xD0, 0x50, 0x4D};
uint32_t devAddr = 0x1100000F;

void RxData(uint8_t* pData, uint8_t dataLength, OpStatus_t status)
{}

void RxJoinResponse(bool status)
{}

void main(void)
{
    // Initialize the device
    SYSTEM_Initialize();

    // If using interrupts in PIC18 High/Low Priority Mode you need to enable
    // the Global High and Low Interrupts
    // If using interrupts in PIC Mid-Range Compatibility Mode you need to
    // enable the Global and Peripheral Interrupts
    // Use the following macros to:

    // Enable high priority global interrupts
    //INTERRUPT_GlobalInterruptHighEnable();

    // Enable low priority global interrupts.
    //INTERRUPT_GlobalInterruptLowEnable();

    // Disable high priority global interrupts
    //INTERRUPT_GlobalInterruptHighDisable();
```


© 2017 Microchip Technology Inc.

例5-4： MAIN.C文件演示代码（续）

```
    // Disable low priority global interrupts.
    //INTERRUPT_GlobalInterruptLowDisable();

    // Enable the Global Interrupts
    INTERRUPT_GlobalInterruptEnable();

    // Enable the Peripheral Interrupts
    INTERRUPT_PeripheralInterruptEnable();

    // Disable the Global Interrupts
    //INTERRUPT_GlobalInterruptDisable();

    // Disable the Peripheral Interrupts
    //INTERRUPT_PeripheralInterruptDisable();

    LORAWAN_Init(RxData, RxJoinResponse);

    LORAWAN_SetNetworkSessionKey(nwkSKey);
    LORAWAN_SetApplicationSessionKey(appSKey);
    LORAWAN_SetDeviceAddress(devAddr);

    LORAWAN_Join(ABP);

    while (1)
    {
        // Add your application code
        LORAWAN_Mainloop();

        // All other function calls of the user-defined
        // application must be made here
        LORAWAN_Send(UNCNF, 2, "LoRa", 4);
    }
}
```

注：

© 2017 Microchip Technology Inc.

全球销售及服务网点

美洲

公司总部 Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 1-480-792-7200
Fax: 1-480-792-7277
技术支持：
http://www.microchip.com/support
网址：www.microchip.com

亚特兰大 Atlanta
Duluth, GA
Tel: 1-678-957-9614
Fax: 1-678-957-1455

奥斯汀 Austin, TX
Tel: 1-512-257-3370

波士顿 Boston
Westborough, MA
Tel: 1-774-760-0087
Fax: 1-774-760-0088

芝加哥 Chicago
Itasca, IL
Tel: 1-630-285-0071
Fax: 1-630-285-0075

达拉斯 Dallas
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律 Detroit
Novi, MI
Tel: 1-248-848-4000

休斯敦 Houston, TX
Tel: 1-281-894-5983

印第安纳波利斯 Indianapolis
Noblesville, IN
Tel: 1-317-773-8323
Fax: 1-317-773-5453
Tel: 1-317-536-2380

洛杉矶 Los Angeles
Mission Viejo, CA
Tel: 1-949-462-9523
Fax: 1-949-462-9608
Tel: 1-951-273-7800

罗利 Raleigh, NC
Tel: 1-919-844-7510

纽约 New York, NY
Tel: 1-631-435-6000

圣何塞 San Jose, CA
Tel: 1-408-735-9110
Tel: 1-408-436-4270

加拿大多伦多 Toronto
Tel: 1-905-695-1980
Fax: 1-905-695-2078

亚太地区

亚太总部 Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

中国 - 北京
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

中国 - 成都
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

中国 - 重庆
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

中国 - 东莞
Tel: 86-769-8702-9880

中国 - 广州
Tel: 86-20-8755-8029

中国 - 杭州
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

中国 - 南京
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

中国 - 青岛
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

中国 - 上海
Tel: 86-21-3326-8000
Fax: 86-21-3326-8021

中国 - 沈阳
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

中国 - 深圳
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

中国 - 武汉
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

中国 - 西安
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

中国 - 厦门
Tel: 86-592-238-8138
Fax: 86-592-238-8130

中国 - 香港特别行政区
Tel: 852-2943-5100
Fax: 852-2401-3431

亚太地区

中国 - 珠海
Tel: 86-756-321-0040
Fax: 86-756-321-0049

台湾地区 - 高雄
Tel: 886-7-213-7830

台湾地区 - 台北
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

台湾地区 - 新竹
Tel: 886-3-5778-366
Fax: 886-3-5770-955

澳大利亚 Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

印度 India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

印度 India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

印度 India - Pune
Tel: 91-20-3019-1500

日本 Japan - Osaka
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

日本 Japan - Tokyo
Tel: 81-3-6880-3770
Fax: 81-3-6880-3771

韩国 Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

韩国 Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 或
82-2-558-5934

马来西亚 Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

马来西亚 Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

菲律宾 Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

新加坡 Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

泰国 Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

欧洲

奥地利 Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

丹麦 Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

芬兰 Finland - Espoo
Tel: 358-9-4520-820

法国 France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

法国 France - Saint Cloud
Tel: 33-1-30-60-70-00

德国 Germany - Garching
Tel: 49-8931-9700

德国 Germany - Haan
Tel: 49-2129-3766400

德国 Germany - Heilbronn
Tel: 49-7131-67-3636

德国 Germany - Karlsruhe
Tel: 49-721-625370

德国 Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

德国 Germany - Rosenheim
Tel: 49-8031-354-560

以色列 Israel - Ra'anana
Tel: 972-9-744-7705

意大利 Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

意大利 Italy - Padova
Tel: 39-049-7625286

荷兰 Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

挪威 Norway - Trondheim
Tel: 47-7289-7561

波兰 Poland - Warsaw
Tel: 48-22-3325737

罗马尼亚 Romania - Bucharest
Tel: 40-21-407-87-50

西班牙 Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

瑞典 Sweden - Gothenberg
Tel: 46-31-704-60-40

瑞典 Sweden - Stockholm
Tel: 46-8-5090-4654

英国 UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820

© 2017 Microchip Technology Inc.