Never stop thinking

PMA71xx/PMA51xx

SmartLEWIS™ MCU

射频发射器 FSK/ASK 315/434/868/915 MHz
内置 8051 微控制器，配 10 位 ADC
内置 125 kHz ASK LF 接收器

应用说明

软件框架
修订版 1.2，2010-06-23
初步修订

感应与控制

版本 2010-06-23
由 Infineon Technologies AG 出版
81726 Munich, Germany

法律免责声明

在任何情况下均不得将本文件所提供的信息视为对条件或特征的担保。 英飞凌科技公司特此声明，对本文中所提及的任何示例或提示、任何典型数值和/或任何与设备应用相关的信息，不作任何及所有形式的担保或承担任何及所有形式的责任（包括但不限于对不侵犯任何第三方知识产权的担保）。

为方便客户浏览，英飞凌以下所提供的将是有关英飞凌产品及服务资料的中文翻译版本。该中文翻译版本仅供参考，并不可作为任何论点之依据。虽然我们尽力提供与英文版本含义一样清楚的中文翻译版本，但因语言翻译和转换过程中的差异，可能存在不尽相同之处。因此，我们同时提供该中文翻译版本的英文版本供您阅读，请参见【PMAx1xx - Software Framework】。并且，我们在此提醒客户，针对同样的英飞凌产品及服务，我们提供更加丰富和详细的英文资料可供客户参考使用。请详见【Transmitter+ Microcontroller】

客户理解并且同意，英飞凌毋须为任何人士由于其在翻译原来的英文版本成为该等中文翻译版本的过程中可能存在的任何不完整或者不准确而产生的全部或者部分、任何直接或者间接损失或损害负责。英飞凌对于中文翻译版本之完整与正确性不担负任何责任。英文版本与中文翻译版本之间若有任何歧异，以英文版本为准，且仅认可英文版本为正式文件。

您如果使用以下提供的资料，则说明您同意并将遵循上述说明。如果您不同意上述说明，请不要使用本资料。

信息

有关技术、交货条款及条件和价格的更多信息，请与您最近的英飞凌科技公司办事处 (**www.infineon.com**) 联系。

警告

由于技术要求，元件可能含有危险物质。 如需相关型号的信息，请与距离您最近的英飞凌科技公司办事处联系。
如果可能合理地预期此类元件的故障会导致生命支持设备或系统发生故障或影响该设备或系统的安全性或有效性，则英飞凌科技公司提供的元件仅可用于获得英飞凌科技公司明确书面批准的生命支持设备或系统。生命支持设备或系统的目的是植入人体或支持和/或保持并维持和/或保护生命。 如果出现故障，则可能危及使用者或他人的健康。

目录

修订历史： 2010-06-23，修订版 1.2

先前版本： 1.1	
页码	**变更内容（自上次修订后的主要变更）**
6	本文件对软件实例修订版 V2.x 适用

Infineon Technologies AG 的商标

ABM™、BlueMoon™、CONVERGATE™、COSIC™、C166™、FALC™、GEMINAX™、GOLDMOS™、ISAC™、OMNITUNE™、OMNIVIA™、PROSOC™、SEROCCO™、SICOFI™、SIEGET™、SMARTi™、SmartLEWIS™、SMINT™、SOCRATES™、VINAX™、VINETIC™、VOIPRO™、X-GOLD™、XMM™、X-PMU™、XWAY™。

其它商标

Microsoft Corporation 的 Microsoft®、Visio®、Windows®、Windows Vista®、Visual Studio®、Win32®。 Linus Torvalds 的 Linux®。 Adobe Systems Incorporated 的 FrameMaker®、Adobe® 、Reader™、Adobe Audition®。 Comneon GmbH & Co OHG 的 APOXI®、COMNEON™。ARM Limited 的 PrimeCell®、RealView®、ARM®、ARM® Developer Suite™ (ADS)、Multi-ICE™、ARM1176JZ-S™、CoreSight™、Embedded Trace Macrocell™ (ETM)、Thumb®、ETM9™、AMBA™、ARM7™、ARM9™、ARM7TDMI-S™、ARM926EJ-S™。 ParthusCeva Inc 的 OakDSPCore®、TeakLite® 、DSP Core, OCEM®。Global Locate 的 IndoorGPS™、GL-20000™、GL-LN-22™。MIPI Alliance 的 mipi™。 DECT Forum 的 CAT-iq™。 MIPS Technologies, Inc 的 MIPS™、MIPS II™、24KEc™、MIPS32®、24KEc™。Texas Instruments Incorporated 的 Texas Instruments®、PowerPAD™、C62x™、C55x™、VLYNQ™、Telogy Software™、TMS320C62x™、Code Composer Studio™、SSI™。Bluetooth SIG, Inc 的 Bluetooth®。The Infrared Data Association 的 IrDA®。 Sun Microsystems, Inc 的 Java™、SunOS™、Solaris™。Koninklijke Philips Electronics N.V. 的 Philips®、I2C-Bus®。Seiko Epson Corporation 的 Epson®。 Kabushiki Kaisha Hattori Seiko Corporation 的 Seiko®。 Matsushita Electric Industrial Co., Ltd. 的 Panasonic®。Murata Manufacturing Company 的 Murata®。 Taiyo Yuden Co., Ltd. 的 Taiyo Yuden™。TDK Electronics Company, Ltd. 的 TDK®。Motorola, Inc. 的 Motorola®。National Semiconductor Corporation 的 National Semiconductor®、MICROWIRE™。 The Institute of Electrical and Electronics Engineers, Inc. 的 IEEE®。Samsung Corporation 的 Samsung®、OneNAND®、UtRAM® 。 Toshiba Corporation 的 Toshiba® 。Dallas Semiconductor Corp 的 Dallas Semiconductor®、1-Wire®。The International Organization for Standardization 的 ISO®。 The International Engineering Consortium 的 IEC™。 EMVCo, LLC 的 EMV™。Zetex Semiconductors 的 Zetex®。 Microtec Research, Inc. 的 Microtec®。Cadence Design Systems, Inc. 的 Verilog®。the American National Standards Institute, Inc. 的 ANSI®。Wind River Systems, Inc. 的 WindRiver® 和 VxWorks®。Mentor Graphics Corporation 的 Nucleus™。 OmniVision Technologies, Inc. 的 OmniVision®。Sharp Corporation 的 Sharp®。 Symbian Software Ltd. 的 Symbian OS®。Openwave Systems, Inc. 的 Openwave®。Maxim Integrated Products, Inc 的 Maxim®。Spansion LLC 的 Spansion®。 Micron Technology, Inc. 的 Micron®、CellularRAM®。RF Micro Devices, Inc. 的 RFMD®。EPCOS AG 的 EPCOS®。 The Open Group 的 UNIX®。 Tektronix, Inc. 的 Tektronix®。Intel Corporation 的 Intel®。 Qimonda AG 的 Qimonda®。Samsung Corporation 的 1GOneNAND®。 Hilgraeve, Inc. 的 HyperTerminal®。The MathWorks, Inc. 的 MATLAB®。Red Hat, Inc. 的 Red Hat®。Cadence Design Systems, Inc. 的 Palladium®。SIRIUS Satellite Radio Inc. 的 SIRIUS Satellite Radio®。TOKO Inc 的 TOKO®。以及 KEIL™。

本文件包含内容如有变更，恕不另行通知。

商标最后更新于 2009-03-10

插图目录

1 引言

本文件以 PMA71xx/PMA51xx 软件框架为例，介绍了 PMA71xx/PMA51xx 软件的构建方法。 可用的软件示例文件包括内联文档，这些文档专为帮助程序员轻松入门和缩短软件面世时间而编制。 您可以通过 **http://www.infineon.com/PMA_tooling**下载软件示例文件。 PMA71xx/PMA51xx 软件框架源代码安装程序是 *PMA_Software_Framework_V2.x.msi*。该安装程序在 *PMA Starter Kit* 和 *PMA Evaluation Kit* 下载包中都能找到。 此外，可以使用 *PMA_Software_Framework_V2.x.msi* 安装程序将用 doxygen 生成的更多源代码文件复制到指定位置。 默认路径为 *PMA_Software_Framework_V2.x/html*。 打开 *index.html* 文件后，在标准浏览器中可以显示这些文件。软件框架可以执行下述功能：

- PMA71xx/PMA51xx 启动文件的实现
- 唤醒与重置处理
- 使用间隔定时器
- 看门狗定时器处理
- 射频传输
- 中断处理

本文介绍了软件框架大致的文件结构、具体特性和推荐的编程流程，与源代码修订版 2.x (*PMA_Software_Framework_V2.x.msi*) 相容。

2 文件结构

本章节概括介绍了 PMA71xx/PMA51xx 软件框架的文件结构，描述了通过源文件实现的功能。 **图 1** 显示各文件之间的关联。

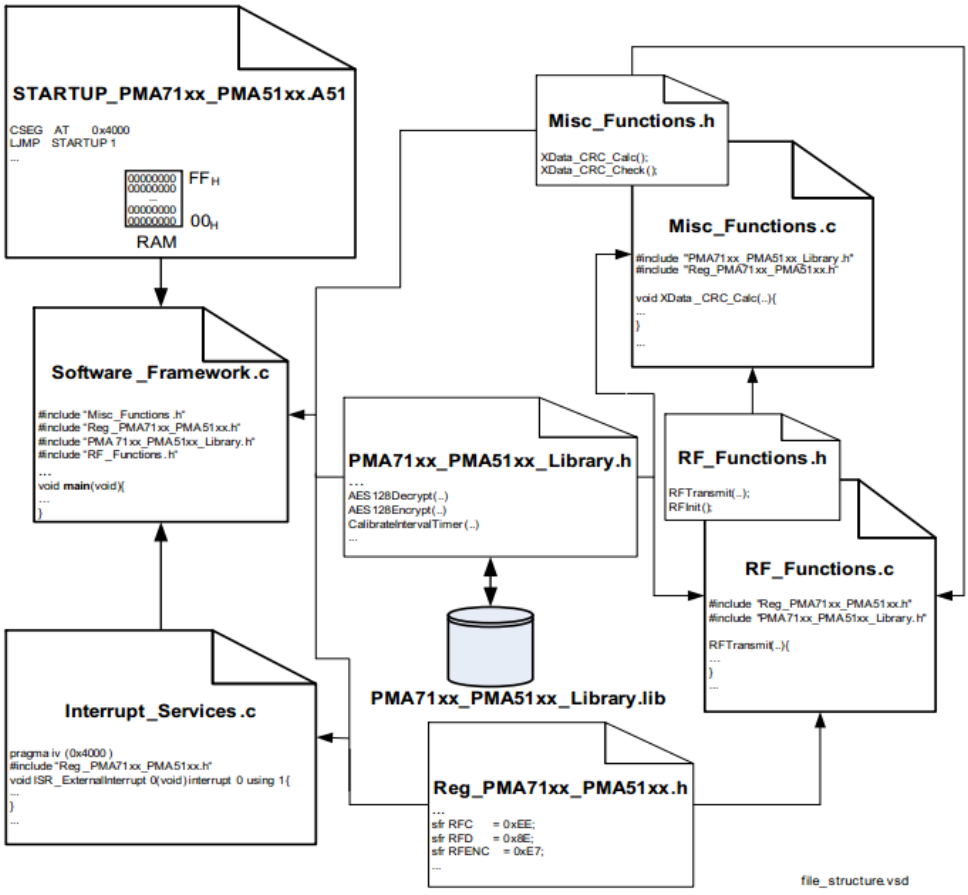

图 1 PMA71xx/PMA51xx 软件框架文件结构

2.1 头文件

头文件中定义了不同模块的接口。

2.1.1 Reg_PMA71xx_PMA51xx.h

这是 PMA71xx / PMA51xx 寄存器定义文件。 文件中定义了 PMA71xx/PMA51xx 所有的特殊功能寄存器 (SFR)。如果需要直接访问SFR，则必须将该文件添加到项目中。

2.1.2 PMA71xx_PMA51xx_Library.h

PMA71xx_PMA51xx_Library.h 是 PMA71xx/PMA51xx 函数库接口。 该文件中定义了函数库的原型和关于射频传输的一些声明。 如果要使用 PMA71xx/PMA51xx 函数库，则必须将该文件和 *PMA71xx_PMA51xx_Library.lib* 一起添加到项目中。

2.1.3 Misc_Functions.h

Xdata_CRC_Calc() 与 *Xdata_CRC_Check()* 函数接口。

2.1.4 RF_Functions.h

RFInit() 与 *RFTransmit()* 函数接口。

2.2 源文件

源文件包括了启动文件、头文件中定义的函数、 *main()* 函数和中断服务例行程序的实现。

2.2.1 STARTUP_PMA71xx_PMA51xx.A51

本文件是标准 8051 启动文件 *STARTUP.A51* （由 KEIL™ 提供）的修订副本，必须添加到项目中。 如果不添加，链接器就会自动添加标准 *STARTUP.A51*。 如果是因重置而启动 PMA71xx/PMA51xx，整个 idata 存储区 00_H – FF_H 都将被初始化为 00_H。 将 SFR 比特位 CFG2.4 [PDLMB] 设为 0，就可以在掉电或热关机模式下为 idata 存储区低位 00_H - $7F_H$ 供电。 如果 PMA71xx/PMA51xx 从掉电或热关机状态醒来，则检查 PDLMB 比特位。 如果在掉电期间或热关机状态时对低位 idata 存储区供电，只有 idata 高位存储块 80_H - FF_H被置0。否则所有 idata 存储区 00_H - FF_H, 都被初始化为 00_H。

最终，堆栈指针被初始化，执行 ljmp 跳转至 *main()* 程序。

2.2.2 Misc_Functions.c

Xdata_CRC_Calc() 与 *Xdata_CRC_Check()* 函数在这个文件中实现。 每个函数的功能描述参见下文。

2.2.2.1 Xdata_CRC_Calc()

使用库函数 *CRC8Calculation()* 计算 xdata 00_H - $0E_H$ 内容的 CRC 校验和并将其存储到 xdata $0F_H$ 中。 因为传到库函数 *CRC8Calculation()* 的缓冲必须是位于存储区空间 idata 中的，所以必须将 xdata 的内容复制到 idta 缓冲中。

2.2.2.2 Xdata_CRC_Check()

检查 xdata 内容的 CRC 校验和是否有效。 从而计算 xdata 整个 00_H - $0F_H$ 内容的 CRC 校验和，校验和必须是 00_H。

2.2.3 Software_Framework.c

本文件包括 PMA71xx/PMA51xx 软件框架的 *main()* 函数，在 *STARTUP_PMA71xx_PMA51xx.A51* 之后执行。 xdata 变量定义于此。 使用 keyword **_at_**，将它们定位到 xdata 存储区的一个绝对位置，因此它们必须采用全局定义。 此外，为了增加代码的可读性，还要对某些比特位进行定义。 *main()* 函数会检查唤醒比特位 DSR.1 [WUP] 以判断 PMA71xx/PMA51xx 是因重置还是因掉电模式时的唤醒事件而启动。

如果端口 1 和 3 的方向被复位，则根据这些引脚的用途设置输出和灵敏度寄存器。 所有端口引脚（而不是 PP2）会被设为输入，内部上拉电阻被启用。 PP2 被设为输入，内部上拉电阻被启用。 这时所有中断都被禁用，但可以通过设置全局中断启用和合适的中断启用位，轻松地启用这些中断。 使用唤醒掩码寄存器 WUM 和 ExtWUM 来启用或禁用内、外唤醒源。 在本软件框架中，热关机内部唤醒和外部唤醒 WU1 (PP2) 被启用（非屏蔽）。 在正常、编程和调试模式中，看门狗 - 和间隔定时器唤醒始终开启。 完成唤醒设置之后，校准间隔定时器。 从而令系统时钟切换到外部石英钟，以获得更高准确度。 在重置例行程序的末尾，会计算 xdata 所有内容的 CRC 校验和。 若 CRC 校验和无效，则 xdata 变量会被（重新）初始化到 0。

如果 PMA71xx/PMA51xx 是从掉电或热关机状态醒来，则检查唤醒源。 若检测到唤醒，则增加合适的 xdata 计数变量，计算 xdata 内容 00_H - $0E_H$ 的 CRC 校验和，然后将其写入 xdata 存储区的 $0F_H$ 中。 如果是看门狗定时器 — 或热关机唤醒，则 PMA71xx/PMA51xx 被复位以确保 SFR 处于定义好的状态。 如果发生间隔定时器或外部 WU1 唤醒，看门狗定时器被重置，射频发射器通过 *RFInit()* 初始化，xdata 变量被复制到 idata 缓冲中，该缓冲通过调用 *RFTransmit()* 进行射频传输。 传输之后，PMA71xx/PMA51xx 被设为掉电模式以减少能耗，等待下个唤醒事件。 所有现有唤醒标志检查已经实现。 为了能够使用其中一个当前被禁用（屏蔽）的唤醒源，必须启用该唤醒源并且处理该唤醒。

2.2.4 Interrupt_Services.c

当前未启用任何中断。 为了能够使用某个中断，必须设置 *Software_Framework.c* 中相应的中断启用位。 在 *Interrupt_Services.c* 中定义了所有中断服务例行程序，这些例行程序在使用中断时必须实现。 这个文件中有很多描述启用每次中断所需设置的比特位的内联文档。

2.2.5 RF_Functions.c

函数 *RFInit()* 与 *RFTransmit()* 在 *RF_Functions.c* 中实现。

2.2.5.1 RFInit()

首先，定义和初始化一个结构体变量*RF_Config* ，参见 *PMA71xx_PMA51xx_Library.h*。 然后，调用库函数 *InitRF()* 为射频传输所需的所有 SFR 设置合适的值。 关于 结构变量*RF_Config* 的每一个元素的详细说明可以参见 **[1]**。

2.2.5.2 RFTransmit()

这个函数实现射频传输。 该函数被用于生成 TDA523x 接收器所需的射频帧。 射频帧详见**第 5 章**。

2.3 函数库文件 (PMA71xx_PMA51xx_Library.lib)

这是实现 *PMA71xx_PMA51xx_* 中定义的所有函数的函数库。

3 PMA71xx/PMA51xx MCU 的特殊性能

PMA71xx/PMA51xx MCU 支持一些 8051 微控制器的标准以外的特殊功能，目的是协助用户进行软件开发。 这些特殊性能如下所示，相关说明参见后面的章节：

- PMA71xx/PMA51xx 函数库
- 唤醒与复位
- 看门狗定时器
- 间隔定时器

3.1 函数库

PMA71xx/PMA51xx 函数库是一组由英飞凌科技开发的、经过优化的函数，可以帮助用户轻松开发 PMA71xx/PMA51xx 软件应用程序。 关于 PMA71xx/PMA51xx 函数库中实现的每个函数的详细说明可以参见 **[1]**。 通过使用 PMA71xx/PMA51xx 函数库函数，用户不需要对 SFR 的每个比特位进行设置就能获得想要的功能。 例如，为了初始化射频传输，需要设置 11 个SFR 中 85 个左右的比特位， 另一方面，只需要调用 PMA71xx/PMA51xx 函数库函数 *InitRF*(..) 就能轻松达到同样的目的，如下所示。

3.1.1 代码示例

为了设置射频发射器，需要初始化定义在 *PMA71xx_PMA51xx_Library.h* 中的 结构变量*RF_Config* ，并且使用该变量作为参数来调用 PMA71xx/PMA51xx 函数库函数 *InitRF(..)*。

```
// 声明和初始化结构变量 RF_Config
struct RF_Config idata myRF_Config;
myRF_Config.Quiescent = 0;
myRF_Config.IntMask = 0;
myRF_Config.DutyControl = DUTY_OFF;
myRF_Config.XcapShort = DISABLE;
myRF_Config.Encoding = MANCHESTER;
myRF_Config.dataLength = 7;
myRF_Config.DutyCycle = DUTY_27;
myRF_Config.Modulation = ASK;
myRF_Config.Invert = NO_INVERT;
myRF_Config.Frequency = RF_315MHZ;
myRF_Config.OutStages = STAGE_3;
myRF_Config.xtal0 = 0x28;
myRF_Config.xtal1 = 0x11;
myRF_Config.baudrate = 9600;

// 调用库函数 InitRF(..)
InitRF(&myRF_Config);
```

3.2 唤醒与复位

唤醒或复位后，PMA71xx/PMA51xx 从代码位置 4000_H 开始执行程序。 为了判断是否有唤醒或复位，可以使用下述代码示例。

3.2.1 代码示例

如果发生了唤醒事件，SFR 比特位 DSR.1 [WUP] 会发出信号。 如果这个比特位没有设置，则 PMA71xx/PMA51xx 只能是从复位启动的。 PMA71xx/PMA51xx 函数库函数 *PowerDown*() 将 PMA71xx/PMA51xx 设为掉电模式以降低能耗。

```
// 导入 PMA71xx/PMA51xx 函数库
#include "PMA71xx_PMA51xx_Library.h"
// 导入 PMA71xx/PMA51xx 寄存器文件（此处定义 DSR）
```

```
#include "Reg_PMA71xx_PMA51xx.h"
#define BIT_WUP 0x02
void main(void){
        // 判断： 复位还是唤醒：
        // ================================
        if ((DSR & BIT_WUP)== 0) {
                // 复位入口：
                // =============
        }
        else {
                // 唤醒入口：
                // ==============
        }
        // 将 PMA71xx/PMA51xx 设定为掉电模式（调用函数库）
        PowerDown();
}
```

3.3 看门狗定时器

为了增加操作的安全性，提供看门狗定时器。 看门狗定时器是一个增序计数器，起始值为 0，通过 2kHz RC-LP 时钟计数。 若看门狗定时器在大约 1 s 之后溢出，唤醒单元会生成一个复位脉冲令系统复位。 从而打破系统不稳定造成的无限软件循环。

看门狗定时器遇到唤醒或空闲事件时会自动复位。

注： 为了避免看门狗定时器复位事件，必须通过设置 SFR 比特位 CFG2.1 [WDRES] 定期复位增序计数器。 看门狗定时器屏蔽位 WUM.7 [WDog] 只有在测试模式下才可用。 在正常模式下，看门狗定时器不可被禁用。 在编程和调试模式下，固件处理程序在其主轮询循环（ I^2C 命令轮询）中复位看门狗定时器。

3.4 间隔定时器

间隔定时器负责在达到指定时间间隔后，将 PMA71xx/PMA51xx 从掉电状态中唤醒。 在正常模式时，间隔定时器和相应的唤醒始终开启。 如**第 3.4.1 章**所示，间隔定时器可以使用 PMA71xx/PMA51xx 函数库函数 *CalibrateIntervalTimer(..)* 轻松配置。

3.4.1 代码示例

在复位输入中，间隔定时器由 PMA71xx/PMA51xx 函数库函数 *CalibrateIntervalTimer(..)* 校准。 为了获得更高的准确度，应使用外部晶体振荡器。

在唤醒输入中，则检查间隔定时器唤醒的唤醒源。 必须执行间隔定时器唤醒处理。

PMA71xx/PMA51xx 函数库函数 *PowerDown*() 将 PMA71xx/PMA51xx 设为掉电模式以降低能耗。

```
// 导入 PMA71xx/PMA51xx 函数库
#include "PMA71xx_PMA51xx_Library.h"
// 导入 PMA71xx/PMA51xx 寄存器文件（此处定义 DSR）
#include "Reg_PMA71xx_PMA51xx.h"
#define BIT_WUP 0x02
#define BIT_IT   0x01

void main(void){
        unsigned long idata Xtal_Frequency_2 = 9040000;
        unsigned char WU_Frequency = 10;

        // 判断： 复位还是唤醒：
        // ================================
        if ((DSR & BIT_WUP)== 0) {
                // 复位入口：
                // =============
```

```
                // 间隔定时器配置：
                // ============================
                // 取决于准确度标准以及时间与电压漂移
                // 推荐重新校准。
                // 从 12MHz 高精度 RC 振荡器切换到外部晶体振荡器
                // 以获得更高准确度。（函数库调用）
                Switch2XTAL();
                // 校准间隔定时器，每 100ms (f = 10Hz) 唤醒 PMA71xx/PMA51xx 一次
                //  （函数库调用）
                CalibrateIntervalTimer(WU_Frequency, &Xtal_Frequency_2);
    }
    else {
                // 唤醒入口：
                // =============
                // 间隔定时器唤醒（注意，WUF 一经读取即被清空！）
                if (WUF & BIT_IT) {
                // .. 处理间隔定时器唤醒
                }
}
    // 将 PMA71xx/PMA51xx 设定为掉电模式（调用函数库）
    PowerDown();
}
```

4 程序流程

第 12 页图2 是软件框架流程图。

PMA71xx/PMA51xx 在启动文件内开始执行程序。 RAM 初始化后，检查 SFR DSR.1 [WUP] 中的唤醒标志，判断 PMA71xx/PMA51xx 是因复位还是因唤醒事件而启动。

如果 PMA71xx/PMA51xx 是因复位而启动，则所有使用的模块被相应的 SFR 初始化。 为了使用外部唤醒，WU1 引脚 PP2 被配置为输入，相应的唤醒被启用。 当前所有中断被禁用，但可以通过使能全局和特定的中断轻松激活。 间隔定时器被校准，计算 XRAM 的CRC 校验和。 如果 XRAM 数据在看门狗定时器或热关机触发复位后依然有效，则使用该校验和进行控制。 最后，PMA71xx/PMA51xx 被设为掉电模式以降低能耗，等待下一个唤醒事件。

如果 PMA71xx/PMA51xx 是因唤醒事件而从掉电模式启动，则检查唤醒源。 软件框架能够处理四种唤醒源， 即看门狗定时器、热关机、间隔定时器和外部 WU1 唤醒。 每次实现唤醒后，就有一个计数变量存入 XRAM 中。 在相应的唤醒服务例行程序中，这些变量会被累加，计算 CRC 校验和。 若发生看门狗定时器和热关机唤醒事件，随后会触发软件复位以确保所有 SFR 处于预设状态。 若发生间隔定时器或外部 WU1 唤醒事件，随后射频发射器被初始化，XRAM 内容通过射频传输出去，PMA71xx/PMA51xx 被设为掉电模式。

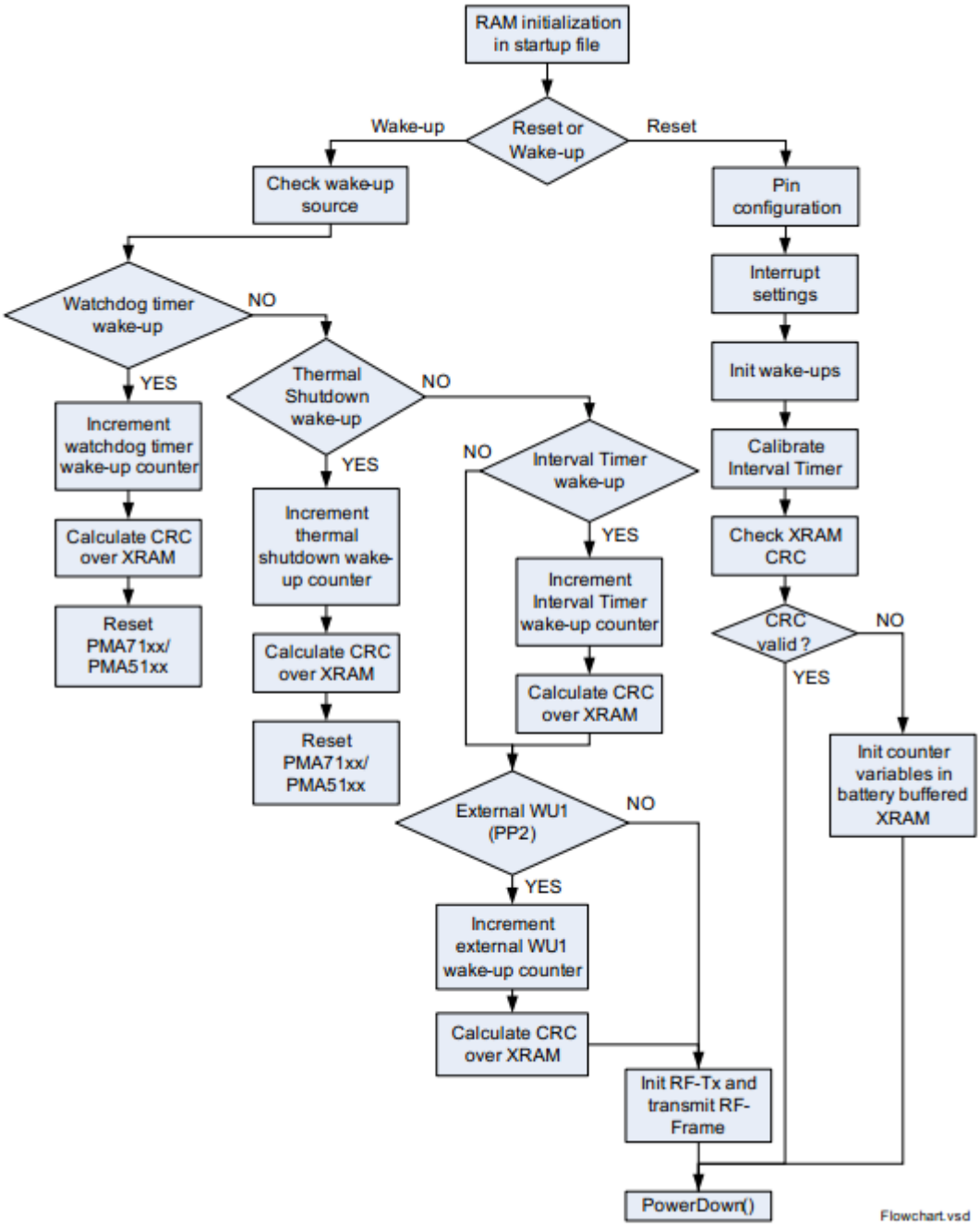

图 2　　PMA71xx/PMA51xx 软件框架流程图

5　射频协议

PMA71xx/PMA51xx 软件框架的射频传输设计与 TDA523x 接收器相容， 因此必须使用特殊的射频协议。

下述设置供射频传输使用（在 *RFInit()* 中设置，参见 *RF_Functions.c*）：

- **编码：** 曼切斯特
- **调制：** FSK
- **波特率** 9600 bps
- **频率：** 434 MHz

射频帧开头是一个由 8 个曼切斯特编码数据位（16 码片）构成的RUNIN 序列，TDA523x 接收器使用这些比特位进行内部滤波器设置和频率调节。 后面接着 16 码片长的 TSI（电报开始标识符），用于同步帧和检测数据帧的确切起始位置（信息净负荷）。 为了检测 EOM（消息结束），会发送一个曼切斯特违例（2 个 1_B 码片）。

5.1 射频传输的信息净负荷

射频传输的信息净负荷为 16 字节长，包含针对各种唤醒的简单型计数器数值。 这些唤醒为看门狗定时器、间隔定时器、热关机和外部 WU1 唤醒。 此外，还有来自 xdata 存储区的一个字节 (*XDB*)，这个字节当前未被使用，但也会通过射频发射传输出去。 每次如有上述四种唤醒事件中的任一事件发生，相应的计数器变量会被实现，射频传输启动。 整个射频帧包括 TDA523x 帧及信息净负荷，参见**第 13 页图 3。**

	RUNIN								TSI								信息净负荷						EOM	
比特位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	WDWU	ITWU	TMWU	EWU1	XDB	CRC		
码片	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	32 位	32 位	32 位	16 位	8 位	8 位	1	1

RUNIN .. 按顺序运行（同步）
TSI .. 电报开始标识符
EWU1 .. 外部 WU1 唤醒计数器
WDWU .. 看门狗定时器唤醒计数器
ITWU .. 间隔定时器唤醒计数器
TMWU .. 热关机唤醒计数器
CRC .. xdata[0H:E H] 校验和
EOM .. 消息结束
XDB .. xdata 字节 **rf_frame.vsd**

图 3 射频帧，包含 TDA523x 帧和信息净负荷

参考资料

[1] 《PMA 函数库指南》

www.infineon.com

由 Infineon Technologies AG 出版