AN2502

应用笔记

UM2006A API 使用指南

版本：V1.0

UNICMICRO

广芯微电子

广芯微电子（广州）股份有限公司

http://www.unicmicro.com/

条款协议

本文档的所有部分，其著作产权归广芯微电子（广州）股份有限公司（以下简称广芯微电子）所有，未经广芯微电子授权许可，任何个人及组织不得复制、转载、仿制本文档的全部或部分组件。本文档没有任何形式的担保、立场表达或其他暗示，若有任何因本文档或其中提及的产品所有资讯所引起的直接或间接损失，广芯微电子及所属员工恕不为其担保任何责任。除此以外，本文档所提到的产品规格及资讯仅供参考，内容亦会随时更新，恕不另行通知。

1. 本文档中所记载的关于电路、软件和其他相关信息仅用于说明半导体产品的操作和应用实例。用户如在设备设计中应用本文档中的电路、软件和相关信息，请自行负责。对于用户或第三方因使用上述电路、软件或信息而遭受的任何损失，广芯微电子不承担任何责任。
2. 在准备本文档所记载的信息的过程中，广芯微电子已尽量做到合理注意，但是，广芯微电子并不保证这些信息都是准确无误的。用户因本文档中所记载的信息的错误或遗漏而遭受的任何损失，广芯微电子不承担任何责任。
3. 对于因使用本文档中的广芯微电子产品或技术信息而造成的侵权行为或因此而侵犯第三方的专利、版权或其他知识产权的行为，广芯微电子不承担任何责任。本文档所记载的内容不应视为对广芯微电子或其他人所有的专利、版权或其他知识产权作出任何明示、默示或其它方式的许可及授权。
4. 使用本文档中记载的广芯微电子产品时，应在广芯微电子指定的范围内，特别是在最大额定值、电源工作电压范围、热辐射特性、安装条件以及其他产品特性的范围内使用。对于在上述指定范围之外使用广芯微电子产品而产生的故障或损失，广芯微电子不承担任何责任。
5. 虽然广芯微电子一直致力于提高广芯微电子产品的质量和可靠性，但是，半导体产品有其自身的具体特性，如一定的故障发生率以及在某些使用条件下会发生故障等。此外，广芯微电子产品均未进行防辐射设计。所以请采取安全保护措施，以避免当广芯微电子产品在发生故障而造成火灾时导致人身事故、伤害或损害的事故。例如进行软硬件安全设计（包括但不限于冗余设计、防火控制以及故障预防等）、适当的老化处理或其他适当的措施等。

目录

1 UM2006A 移植

UM2006A 的示例代码已封装相关驱动接口，把 UM2006A 文件夹的所有文件拷贝到项目工程文件下，在移植过程中仅需要移植实现 um2006A_hal 硬件接口，其他文件无需修改。

注：所有的示例代码下的 UM2006A 驱动文件接口都一样，仅可能存在 rfconfig.h 不一样，rfconfig.h 文件可由 RFOCT 上位机导出，可直接替换。

1.1 文件准备

1. UM2006A 驱动文件

   路径：/ UM2006A/*

   描述：UM2006A 文件夹包含以下文件：

   - radio.h ：包括初始化、接收数据包
   - radio.c ：实现初始化、接收数据包
   - um2006A.h ：UM2006A 驱动文件接口
   - um2006A.c ：实现 UM2006A 驱动
   - rfconfig.h ：配置文件，修改配置直接替换，由上位机 RFOCT 导出
   - typedefs.h ：类型定义
   - um2006A_defs.h ：UM2006A 的宏定义
   - um2006A_hal.h ：UM2006A 硬件接口
   - um2006A_hal.c ：UM2006A 硬件接口实现

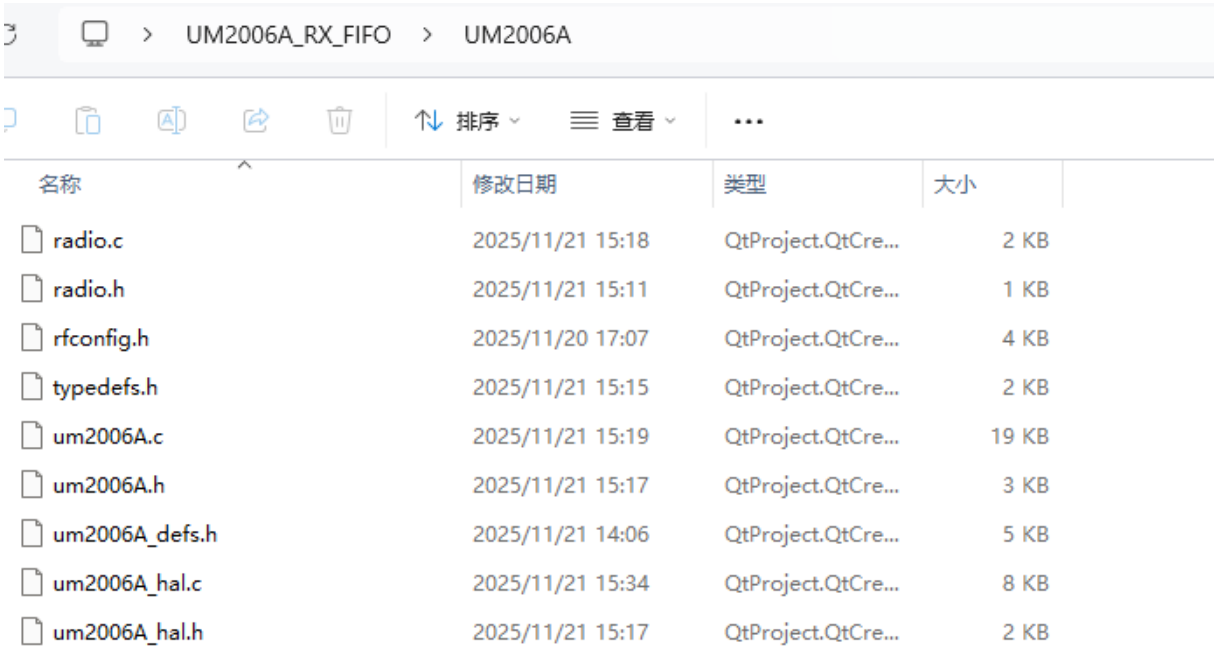

图 1-1：UM2006A 驱动文件夹的文件

1.2 硬件接口

1.2.1 um2006A_hal 介绍

移植文件（um2006A_hal）的硬件接口介绍：

1. void um2006A_hal_init(void)

   硬件初始化，实现以下功能：

   - 实现配置中断输出连接的 GPIO 初始化，并使能中断，采用单边沿的上升沿中断。
   - 实现控制 SDN 的 GPIO 的初始化功能，并拉低 SDN 启动让芯片处于工作中（SDN 可下拉接地，不需要配置）。

2. void um2006A_hal_delay_us (u16_t us)

   UM2006A 延时，实现以下功能：

   - 延迟函数，实现μs 延时。

3. void spi_init(void)

SPI 初始化，实现以下功能：

- SPI 的 IO 的硬件初始化，采用模拟三线 SPI，需初始化与芯片 CS、CLK、SDA 连接 IO 的初始化。

4. void spi_cs_enable(void)

   SPI CS 使能，实现以下功能：

   - 实现 CS 的拉低功能。

5. void spi_cs_disable(void)

   SPI CS 不使能，实现以下功能：

   - 实现 CS 的拉高功能。

6. void spi_write_byte(u8_t byte)

   SPI 写一个字节，实现以下功能：

   - 实现 SPI 时序写一个字节。

7. u8_t spi_read_byte(void)

   SPI 读一个字节，实现以下功能：

   - 实现 SPI 时序读一个字节。

1.2.2 um2006A_hal 移植

UM2006A 的移植仅需要移植 um2006A_hal 文件，um2006A_hal 移植过程中仅需要实现以下功能：

1. 宏定义

```
#define UM2006A_HAL_SPI_CS_ENABLE          // 使能 CS
#define UM2006A_HAL_SPI_CS_DISABLE         // 不使能 CS
#define UM2006A_HAL_SPI_CLK_LOW            // 拉低时钟
#define UM2006A_HAL_SPI_CLK_HIGH           // 拉高时钟
#define UM2006A_HAL_SPI_DATA_LOW           // 拉低数据引脚
```



```
#define UM2006A_HAL_SPI_DATA_HIGH                // 拉高数据引脚
#define UM2006A_HAL_SPI_DATA_GET                 // 获取数据引脚电平
#define UM2006A_HAL_SPI_DATA_INPUT               // 设置数据引脚输入
#define UM2006A_HAL_SPI_DATA_OUTPUT              // 设置数据引脚输出
#define UM2006A_HAL_SPI_DELAY                    // SPI 时序延时
#define UM2006A_HAL_DELAY_US(us)                 // US 延时
```

2. 硬件初始化

```
void um2006A_hal_init(void)
{
    /* 接收完成中断的 GPIO 设置为输入下拉，单边上升沿，中断调用 um2006A_hal_irq */
    /* SDN 下拉接地，无需配置 */
}
```

3. SPI 初始化

```
void spi_init(void)
{
    /* CS 输出，默认输出高电平 */
    /* CLK 输出，默认输出低电平 */
    /* SDA 输出，使能输入，默认输出高电平 */
}
```

1.3 RX 流程

UM2006A 接收数据流程:

- radio_init 初始化
- um2006A_into_rx 进入接收
- um2006A_clear_all_valid 清除中断标志
- radio_recv_data 接收数据

详细接收流程图如下:

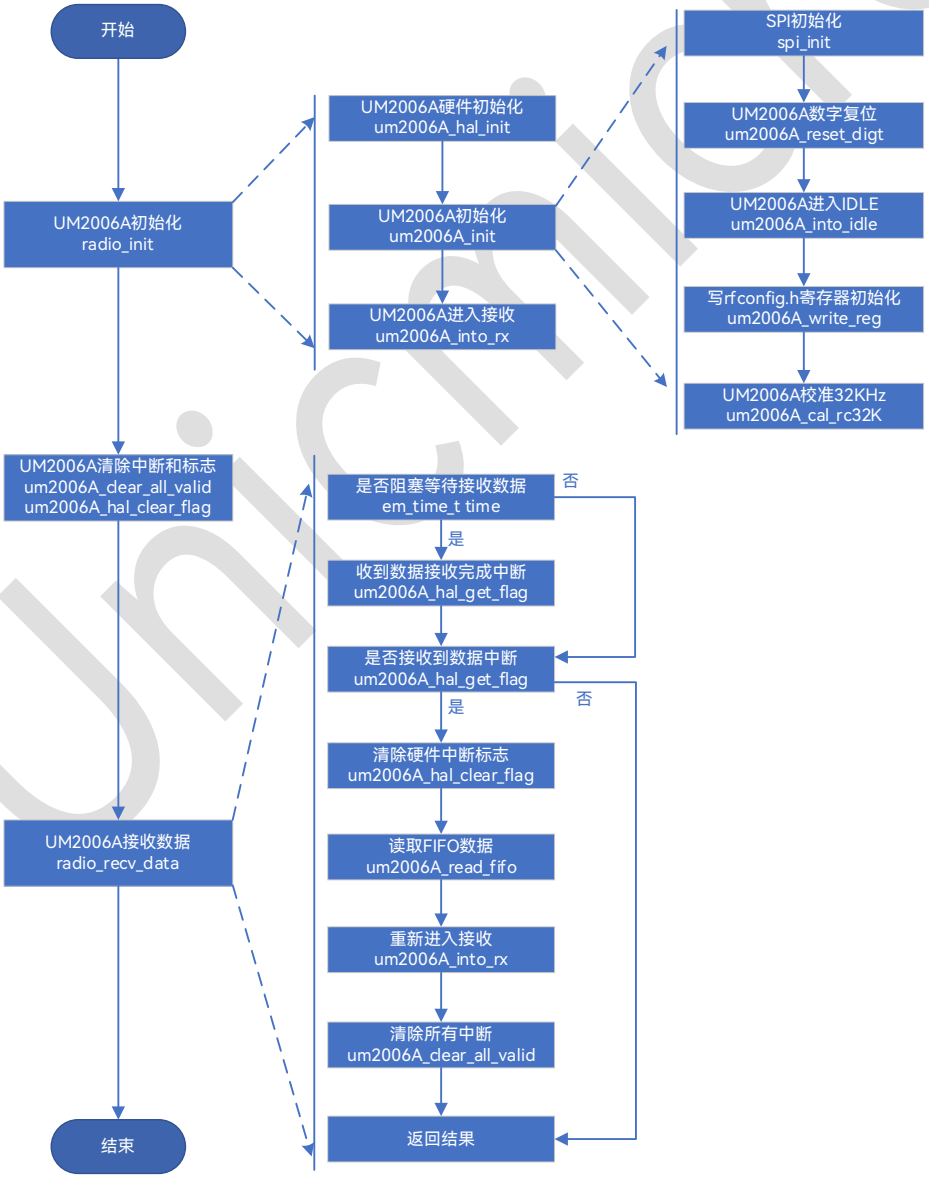

图 1-2：接收数据流程图

2 Radio 接口定义

#include “radio.h”

2.1 radio_init

函数功能	radio_init
函数描述	Radio 初始化
函数定义	em_ret_t radio_init(void)
输入参数	None
输出参数	None
函数返回	em_ret_t 返回初始化结果： ➢ RET_OK：返回成功 ➢ RET_ERR：返回失败

2.2 radio_recv_data

函数功能	radio_recv_data
函数描述	radio 接收一包数据
函数定义	em_ret_t radio_recv_data(u8_t *dat,u8_t *len,um_time_t time)
输入参数	um_time_t time：是否阻塞接收数据 ➢ TIME_WAITTING_FOREVER：阻塞方式 ➢ TIME_WAITTING_NO：非阻塞方式
输出参数	u8_t *dat：接收到的数据 u8_t *len：接收到数据的长度
函数返回	em_ret_t 返回初始化结果： ➢ RET_OK：返回成功 ➢ RET_ERR：返回失败

3 UM2006A 接口定义

#include “um2006A.h”

3.1 um2006A_init

函数功能	um2006A_init
函数描述	初始化 UM2006A
函数定义	void um2006A_init(void)
输入参数	None
输出参数	None
函数返回	None

3.2 um2006A_write_reg

函数功能	um2006A_write_reg
函数描述	写寄存器
函数定义	void um2006A_write_reg(u8_t addr,u8_t value)
输入参数	u8_t addr：寄存器地址 u8_t value：寄存器值
输出参数	None
函数返回	None

3.3 um2006A_write_regs

函数功能	um2006A_write_regs
函数描述	写连续地址寄存器
函数定义	void um2006A_write_regs(u8_t addr,u8_t *dat,u8_t len)
输入参数	u8_t addr：起始寄存器地址 u8_t *dat：连续寄存器值 u8_t len：写寄存器长度
输出参数	None
函数返回	None

3.4 um2006A_read_reg

函数功能	um2006A_read_reg
函数描述	读寄存器
函数定义	u8_t um2006A_read_reg(u8_t addr)
输入参数	u8_t addr：寄存器地址
输出参数	None
函数返回	u8_t 返回寄存器值

3.5 um2006A_read_regs

函数功能	um2006A_read_regs
函数描述	读连续地址寄存器
函数定义	u8_t um2006A_read_regs(u8_t addr,u8_t *dat,u8_t len)
输入参数	u8_t addr：读取起始地址 u8_t len：读取长度
输出参数	u8_t *dat：寄存器值
函数返回	u8_t：返回读取寄存器长度

3.6 um2006A_read_fifo

函数功能	um2006A_read_fifo
函数描述	读 FIFO 数据
函数定义	void um2006A_read_fifo(u8_t *dat,u8_t len)
输入参数	u8_t len：读 FIFO 长度（len<=32）
输出参数	u8_t *dat : FIFO 数据
函数返回	None

3.7 um2006A_into_idle

函数功能	um2006A_into_idle
函数描述	进入 IDLE 模式
函数定义	void um2006A_into_idle(void)
输入参数	None
输出参数	None
函数返回	None

3.8 um2006A_into_sleep

函数功能	um2006A_into_sleep
函数描述	进入 Sleep 模式
函数定义	void um2006A_into_sleep(void)
输入参数	None
输出参数	None
函数返回	None

3.9 um2006A_into_rx

函数功能	radio_init
函数描述	进入 RX 模式
函数定义	void um2006A_into_rx(void)
输入参数	None
输出参数	None
函数返回	None

3.10 um2006A_cal_rc32K

函数功能	um2006A_cal_rc32K
函数描述	校准 32K
函数定义	void um2006A_cal_rc32K(void)
输入参数	None
输出参数	None
函数返回	None

3.11 um2006A_reset_digit

函数功能	um2006A_reset_digit
函数描述	数字复位
函数定义	void um2006A_reset_digit(void)
输入参数	None
输出参数	None
函数返回	None

3.12 um2006A_set_freq

函数功能	um2006A_set_freq
函数描述	设置频点
函数定义	void um2006A_set_freq(double rf_freq,double ref_freq)
输入参数	double rf_freq：频点，单位 MHz double ref_freq：晶振，单位 MHz
输出参数	None
函数返回	None

3.13 um2006A_set_modulation

函数功能	um2006A_set_modulation
函数描述	设置调制方式
函数定义	void um2006A_set_modulation(em_mod_t mod)
输入参数	em_mod_t mod：调制方式 ➢ OOK: OOK 调制方式 ➢ FSK：FSK 和 GFSK 调制方式
输出参数	None
函数返回	None

3.14 um2006A_set_packet_mode

函数功能	um2006A_set_packet_mode
函数描述	设置数据模式
函数定义	void um2006A_set_packet_mode(em_packet_t mode);
输入参数	em_packet_t mode：数据模型 ➢ DIRECT：直通输出模式 ➢ PACKET：数据包模式
输出参数	None
函数返回	None

3.15 um2006A_set_preamble_match_length

函数功能	um2006A_set_preamble_match_length
函数描述	设置 Preamble 匹配长度
函数定义	void um2006A_set_preamble_match_length(u8_t len)
输入参数	u8_t len：长度
输出参数	None
函数返回	None

3.16 um2006A_set_syncword

函数功能	um2006A_set_syncword
函数描述	设置同步字
函数定义	void um2006A_set_syncword(u32_t dat,em_syncword_len_t len)
输入参数	u32_t dat：同步字数据 em_syncword_len_t len：同步字长度 ➢ SYNCWORD_LEN_16BIT：16bit 同步字 ➢ SYNCWORD_LEN_32BIT：32bit 同步字
输出参数	None
函数返回	None

3.17 um2006A_set_payload_length

函数功能	um2006A_set_payload_length
函数描述	设置 Payload 长度
函数定义	void um2006A_set_payload_length(u8_t length)
输入参数	u8_t length：数据长度
输出参数	None
函数返回	None

3.18 um2006A_set_uart_enable

函数功能	um2006A_set_uart_enable
函数描述	使能串口输出
函数定义	void um2006A_set_uart_enable(BOOL enable)
输入参数	BOOL enable：使能选择 ➢ TRUE：不使能 ➢ FALSE：使能
输出参数	None
函数返回	None

3.19 um2006A_set_manchester_mode

函数功能	um2006A_set_manchester_mode
函数描述	设置 Payload 数据的编码方式
函数定义	void um2006A_set_manchester_mode(em_manchst_mode_t mode)
输入参数	em_manchst_mode_t mode：编码方式 ➢ NONE：无编码 ➢ HIGH_01_LOW_10：上升沿为 1，下降沿为 0 ➢ HIGH_10_LOW_01：上升沿为 0，下降沿为 1 ➢ DIFFERENTIAL：差分曼彻斯特编码
输出参数	None
函数返回	None

3.20 um2006A_set_sdo_sel

函数功能	um2006A_set_sdo_sel
函数描述	设置 SDO 输出
函数定义	void um2006A_set_sdo_sel(em_gpio_sel sdo_sel)
输入参数	em_gpio_sel sdo_sel：SDO 输出选择 ➢ GPIO_DEMOD_BITDATA：输出直通数据 ➢ GPIO_CLK_OUT：输出时钟 ➢ GPIO_RXBIT_DATA：输出解调的数据 ➢ GPIO_RXBYTE_DONE：接收完成中断 ➢ GPIO_RSSI_VALID：RSSI 有效 ➢ GPIO_PJD_VALID：PJD 有效 ➢ GPIO_SYNC_VALID：同步字有效 ➢ GPIO_RXBYTE_EN：接收字状态输出 ➢ GPIO_UART_TXD：串口输出 ➢ GPIO_WOR_EVENT：WOR 输出 ➢ GPIO_SPI_MISO：SPI 的 MISO ➢ GPIO_RX_EN：接收状态输出 ➢ GPIO_LIM_I：I 路输出 ➢ GPIO_LIM_Q：Q 路输出 ➢ GPIO_HIGH_LEVEL：高电平输出 ➢ GPIO_LOW_LEVEL：低电平输出
输出参数	None
函数返回	None

3.21 um2006A_set_gpio0_sel

函数功能	um2006A_set_gpio0_sel
函数描述	设置 GPIO0 输出
函数定义	void um2006A_set_gpio0_sel(em_gpio_sel gpio0_sel)
输入参数	em_gpio_sel gpio0_sel：GPIO0 输出选择 ➢ GPIO_DEMOD_BITDATA：输出直通数据 ➢ GPIO_CLK_OUT：输出时钟 ➢ GPIO_RXBIT_DATA：输出解调的数据 ➢ GPIO_RXBYTE_DONE：接收完成中断 ➢ GPIO_RSSI_VALID：RSSI 有效

	➢ GPIO_PJD_VALID：PJD 有效 ➢ GPIO_SYNC_VALID：同步字有效 ➢ GPIO_RXBYTE_EN：接收字状态输出 ➢ GPIO_UART_TXD：串口输出 ➢ GPIO_WOR_EVENT：WOR 输出 ➢ GPIO_SPI_MISO：SPI 的 MISO ➢ GPIO_RX_EN：接收状态输出 ➢ GPIO_LIM_I：I 路输出 ➢ GPIO_LIM_Q：Q 路输出 ➢ GPIO_HIGH_LEVEL：高电平输出 ➢ GPIO_LOW_LEVEL：低电平输出
输出参数	None
函数返回	None

3.22 um2006A_set_gpio1_sel

函数功能	um2006A_set_gpio1_sel
函数描述	设置 GPIO1 输出
函数定义	void um2006A_set_gpio1_sel(em_gpio_sel gpio1_sel)
输入参数	em_gpio_sel gpio1_sel：GPIO1 输出选择 ➢ GPIO_DEMOD_BITDATA：输出直通数据 ➢ GPIO_CLK_OUT：输出时钟 ➢ GPIO_RXBIT_DATA：输出解调的数据 ➢ GPIO_RXBYTE_DONE：接收完成中断 ➢ GPIO_RSSI_VALID：RSSI 有效 ➢ GPIO_PJD_VALID：PJD 有效 ➢ GPIO_SYNC_VALID：同步字有效 ➢ GPIO_RXBYTE_EN：接收字状态输出 ➢ GPIO_UART_TXD：串口输出 ➢ GPIO_WOR_EVENT：WOR 输出 ➢ GPIO_SPI_MISO：SPI 的 MISO ➢ GPIO_RX_EN：接收状态输出 ➢ GPIO_LIM_I：I 路输出 ➢ GPIO_LIM_Q：Q 路输出 ➢ GPIO_HIGH_LEVEL：高电平输出 ➢ GPIO_LOW_LEVEL：低电平输出

输出参数	None
函数返回	None

3.23 um2006A_set_clkout

函数功能	um2006A_set_clkout
函数描述	时钟输出
函数定义	void um2006A_set_clkout(em_clkout_t clk)
输入参数	em_clkout_t clk：时钟输出 ➢ RXBIT_CLK：RX 解调时钟输出 ➢ DEMOD_OUT_CE：直通时钟输出 ➢ CLK_ADC：ADC 时钟输出 ➢ CLK_32K：RC32K 输出
输出参数	None
函数返回	None

3.24 um2006A_set_mute_mode

函数功能	um2006A_set_mute_mode
函数描述	直通输出静音选择
函数定义	void um2006A_set_mute_mode(em_mute_t mode)
输入参数	GPIO 直通输出的静音选择： ➢ NONE_VALID：不屏蔽输出 ➢ RSSI_VALID：RSSI 有效输出 ➢ PJD_VALID：PJD 有效输出 ➢ SYNC_VALID：同步字有效输出
输出参数	None
函数返回	None

3.25 um2006A_read_rssi

函数功能	um2006A_read_rssi
函数描述	读取 RSSI 值
函数定义	float um2006A_read_rssi(void)
输入参数	None
输出参数	None
函数返回	float：返回 RSSI 值

3.26 um2006A_clear_all_valid

函数功能	um2006A_clear_all_valid
函数描述	清除所有中断标志
函数定义	void um2006A_clear_all_valid(void)
输入参数	None
输出参数	None
函数返回	None

3.27 um2006A_get_pjd_valid

函数功能	um2006A_get_pjd_valid
函数描述	获取 PJD 中断标志
函数定义	u8_t um2006A_get_pjd_valid(void)
输入参数	None
输出参数	None
函数返回	u8_t 返回 PJD 中断标志： ➢ 0：无中断 ➢ 1：产生中断

3.28 um2006A_get_rssi_valid

函数功能	um2006A_get_rssi_valid
函数描述	获取 RSSI 的中断标志
函数定义	u8_t um2006A_get_rssi_valid(void)
输入参数	None
输出参数	None
函数返回	u8_t 返回 RSSI 中断标志: ➢ 0：无中断 ➢ 1：产生中断

3.29 um2006A_get_sync_valid

函数功能	um2006A_get_sync_valid
函数描述	同步字中断标志
函数定义	u8_t um2006A_get_sync_valid(void)
输入参数	None
输出参数	None
函数返回	u8_t 返回同步字中断标志: ➢ 0：无中断 ➢ 1：产生中断

3.30 um2006A_get_rxbyte_done

函数功能	um2006A_get_rxbyte_done
函数描述	获取接收完成中断
函数定义	u8_t um2006A_get_rxbyte_done(void)
输入参数	None
输出参数	None
函数返回	u8_t 返回接收完成中断标志: ➢ 0：无中断 ➢ 1：产生中断

4 UM2006A 硬件接口

#include “um2006A_hal.h”

4.1 um2006A_hal_get_flag

函数功能	um2006A_hal_get_flag
函数描述	获取中断标志
函数定义	u8_t um2006A_hal_get_flag(void)
输入参数	None
输出参数	None
函数返回	u8_t 返回中断标志: ➢ 0：无中断产生 ➢ 1：已产生中断

4.2 um2006A_hal_clear_flag

函数功能	um2006A_hal_clear_flag
函数描述	清除中断标志
函数定义	void um2006A_hal_clear_flag(void)
输入参数	None
输出参数	None
函数返回	None

4.3 um2006A_hal_delay_us

函数功能	um2006A_hal_delay_us
函数描述	US 延时
函数定义	void um2006A_hal_delay_us(u16_t us)
输入参数	u16_t us：微秒延时
输出参数	None
函数返回	None

4.4 um2006A_hal_init

函数功能	um2006A_hal_init
函数描述	硬件初始化
函数定义	u8_t um2006A_hal_init(void)
输入参数	None
输出参数	None
函数返回	None

4.5 spi_init

函数功能	spi_init
函数描述	SPI 初始化
函数定义	void spi_init(void)
输入参数	None
输出参数	None
函数返回	None

4.6 spi_cs_enable

函数功能	spi_cs_enable
函数描述	CS 使能
函数定义	void spi_cs_enable(void)
输入参数	None
输出参数	None
函数返回	None

4.7 spi_cs_disable

函数功能	spi_cs_disable
函数描述	CS 不使能
函数定义	void spi_cs_disable(void)
输入参数	None
输出参数	None
函数返回	None

4.8 spi_write_byte

函数功能	spi_write_byte
函数描述	SPI 写一个字节
函数定义	void spi_write_byte(u8_t byte)
输入参数	u8_t byte：写入数据
输出参数	None
函数返回	None

4.9 spi_read_byte

函数功能	spi_read_byte
函数描述	SPI 读一个字节
函数定义	u8_t spi_read_byte(void)
输入参数	None
输出参数	None
函数返回	u8_t：返回读取数据

5 版本修订

版本	日期	描述
V1.0	2025.10.15	初始版