AN1703

应用笔记

UM2020 提升可靠性使用指南

版本：V1.0

广芯微电子（广州）股份有限公司

http://www.unicmicro.com/

条款协议

目录

1. 摘要

本篇应用笔记主要介绍UM2020提升可靠性使用指南。

本篇应用笔记主要包括：

- 概述
- 操作原理
- 操作流程
- 发射协议
- 接收端示例代码

注：具体功能及寄存器的操作等相关事项请以用户手册为准。

2. 概述

本文介绍如何通过芯片自动超时和协议来提升芯片的唤醒成功率和提高芯片的抗干扰能力。

3. 操作原理

提高芯片唤醒成功率是通过芯片自动超时使芯片重新进入监听模式，使芯片处于监听的最佳状态。如芯片正在解码数据时，自动超时芯片重新进入监听状态会导致当前解码的数据丢失，可通过协议来完善，让芯片重新可以接收到数据。当自动超时导致芯片接收数据丢失时，第二帧的数据必须保证能正常接收。

4. 操作流程

4.1 芯片配置优化

芯片需要进行如下配置优化:

1. {0, 0x75}， // 配置芯片工作在标准监听模式。
2. {1, 0x14}， // 要使能 3dB 增益，并关闭内部天线阻尼。
3. {15, 0x9A}，// 自动超时时间设置成 50ms。
4. {27, 0x00}，// 关闭自动调谐。

4.2 流程

1. 初始化 UM2020，并写入配置。
2. 进入监听模式,寄存器 0x7F 写入 0x03。
3. 芯片被唤醒后，首先需要先关闭超时。
4. 等待芯片接收完成数据后读取数据。
5. 芯片接收完成数据后，开启自动超时功能。
6. 清除唤醒状态并重新进入监听状态。

5. 发射协议

为了防止数据因自动超时重新进入监听导致数据丢失，发射需要严格按照协议发射，完整一帧协议图如下：

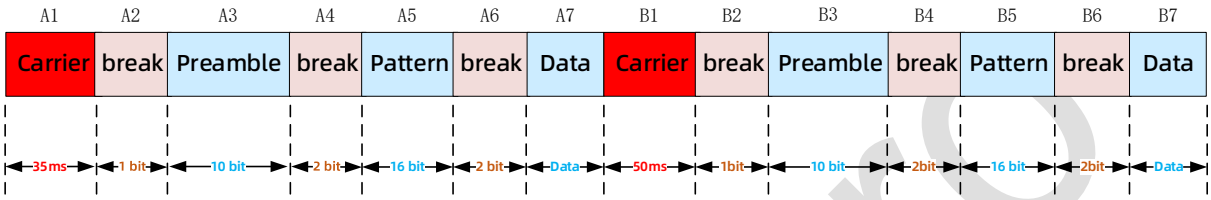

图 5-1：发射协议图

假设接收超时设置为 50ms，发射 1bit 时间为 t, Data 长度为 m（byte），数据曼彻斯特编码。协议要求如下：

1. A1 时间 > A2~A6 的时间和（超时设置 50ms，速率建议大于 2kbps）。
2. A1 时间 > 50 – (A2~A6 的时间总和)。
3. B1 时间 > 50n-16mt 且 B1 时间>50-A1 时间（n 取满足要求的最小正整数）。

注：A1 和 B1 时间建议上浮 5%~10%之间。

如：接收超时时间设置为 50ms，速率采用 2.4kbps，即 1bit 时间 t = 0.416ms，Data 长度为 8bytes，数据曼彻斯特编码：

1. A2~A6 的时间总和 = A2~A6 共 31bits = 31 * 0.416 ms ≈ 13 ms。
2. A1 时间 = 50 – (A2~A6 时间总和) = 50 ms – 13 ms ≈ 37 ms。
3. B1 时间 = 50 * n - 16 * 8 * 0.416 = 50 * n – 54 ms ≈ 46 ms（因 B1 需要大于 50 - A1 = 13 ms，所以 n 取 2）。

6. 接收端示例代码

6.1 芯片配置（2.4kbps）

```
const unsigned char lf_config[][2] =
{
    {0,0x75},
    {1,0x14},
    {2,0x71},
    {3,0x01},
    {12,0x00},
    {14,0x0C},
    {15,0x9A},
    {16,0xAC},
    {21,0x0D},
    {22,0x3F},
    {27,0x00},
    {28,0x33},
    {29,0x33},
    {30,0x33},
    {41,0x23},
    {42,0x33},
    {43,0x01},
};
```

6.2 芯片进入监听

```
void lf_start_listen(void)
{
    lf_write_reg(0x7F,0x03);
}
```

6.3 清除监听状态

```
void lf_clear_wakeup(void)
{
    lf_write_reg(0x15,0x0C);   /* 清除唤醒状态，返回监听模式，RSSI 检测复位 */
}
```

6.4 芯片初始化

```
uint8_t lf_init()
{
    reg_val = lf_read_reg(0x15);
     lf_write_reg(0x15,reg_val | 0x02);                    /* 对 RSSI 检测进行复位 */

    len = sizeof(lf_config)/sizeof(lf_config[0]);
    for(i=0;i<len;i++)
    {
        lf_write_reg(lf_config[i][0],lf_config[i][1]);
    }

    lf_start_listen ();
    return 0;
}
```

6.5 芯片唤醒后接收数据

```
if (lf_wake_flag)
{
    lf_wake_flag = faslse;

    /* 关闭自动超时 */
    reg0F = lf_read_reg(0x0F);
    lf_write_reg(0x0F,(_ reg0F & 0x0F));                  // 关闭复位

    /* 等待接收数据 */
    delay_ms(60);
```

```
    /* 读取数据 */
    lf_read_data(data,8);

    /* 恢复自动超时 */
    lf_write_reg(0x0F,reg0F);
    lf_clear_wakeup();
}
```

7. 版本修订

版本	日期	描述
V1.0	2025.05.20	初始版