Shanghai Fudan Microelectronics Group Company Limited

复旦微电子

FM33G0XX
低功耗系列 MCU
应用笔记

LPUART 低功耗串口

AN004 **V1.0**

联系方式：

电表产品应用：

邢杰： xingjie@fmsh.com.cn TEL：13916427310

陈钊： chenzhao@fmsh.com.cn TEL：18616125501

水气热表及智能家居：

朱发旺：zhufawang@fmsh.com.cn TEL：17749796664

姜涛： jiangtao@fmsh.com.cn TEL：18701992908

超高频 900M 及物联网相关：

王晓腾：wangxiaoteng@fmsh.com.cn TEL：13585663727

王天纵：wangtianzong@fmsh.com.cn TEL：18221803903

资料下载及交流：

开发者论坛：http://www.fmdevelopers.com.cn

目　　录

图目录

表目录

1 说明

本文档为 FM33G0XX 系列低功耗 MCU 的应用笔记，用于说明低功耗 LPUART 的使用方法和普通 UART 模拟 LPUART 的方法。FM33G0XX 系列是复旦微电子公司开发的低功耗 MCU 芯片，请联系复旦微电子公司提供更多相关文档支持设计开发。

2 概述

LPUART 是 FM33G0XX 特有的低功耗串口，其工作时钟仅需 32768Hz 时钟，可支持 300~9600 的波特率的数据接收。LPUART 功耗极低约为 80nA，支持 Sleep/DeepSleep 模式下的数据收发。

2.1 使用说明

1、外设时钟设置使能

☐LPUFCKEN（功能时钟）　　☐LPUARTCKEN（寄存器总线时钟）。

2、管脚配置

LPUART 和 UART0 复用相同的 IO 口

在配置 IO 时需要先配置 PF4 功能选择寄存器 PF4AFSEL，将 PF4 的复用功能选择为 LPUART _TX。

LPUART_RX 与 UART0_RX 相同，不存在复用功能寄存器，若外部不存在上拉，只需要把 PF3 配置成数字功能上拉即可。

3、波特率设置

以波特率 9600 为例，LPUART 使用的主时钟为 32768HZ，其分频与 9600 相差较大。所以采用 3、4 分频交替使用。从波形上看每个 bit 的长度不是 104.16us,而是 91.55us 和 122.07us 的交替。如图 2-1 所示

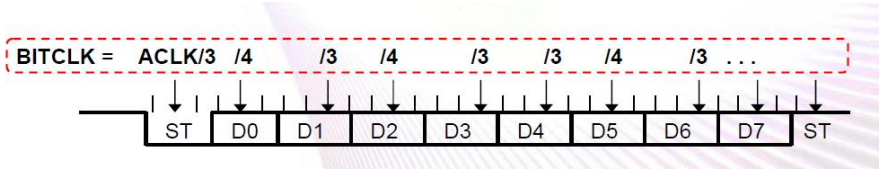


图 2-1 主时钟 3、4 分频交替

为了实现上述的功能，LPUART 在配置波特率寄存器 LPUBAUD 的同时，还需按表 2-1 所示配置 MCTL 寄存器。

表 2-1 配置 MCTL 寄存器

Baud	MCTL											
	Bit0（start）	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11
9600	0	1	0	0	1	0	1	0	1	0	0	1
4800	1	1	0	1	1	1	1	1	0	1	1	1
2400	1	1	0	1	1	0	1	1	0	1	1	0
1200	0	1	0	0	1	0	0	1	0	0	1	0
600	0	1	1	0	1	0	1	1	0	1	1	0
300	0	1	0	0	0	0	1	0	0	0	0	1

4、配置 LPUCON 控制寄存器

在 LPUCON 寄存器中可以设置串口的一些通信参数。特别说明 RXEV。RXEV——接收中断事件配置，用于控制何种事件下向 CPU 提供接收中断。有四种事件分别为：

●START 位检测唤醒。对应的标志位为 LPUSTA 的 bit5 START。

●1byte 数据接收完成。对应的标志位为 LPUIF 的 bit0 RXIF。

●接收数据匹配成功。对应的标志位 LPUSTA 的 bit0 MATCH。

●RXD 下降沿唤醒。对应的标志位为 LPUIF 的 bit1 RXNEGIF。

其中，1byte 数据接收完成和接收数据匹配成功，模块只判断数据位，不判断校验和停止位。示例中具体 LPUART_Init 函数如图 2-2 所示

```
void LPUart_Init(void)
{
  LPUART_SInitTypeDef init_para;

  RCC_PERCLK_SetableEx(LPUFCKEN, ENABLE); //LPUART功能时钟使能
  RCC_PERCLK_SetableEx(LPUARTCKEN, ENABLE); //LPUART寄存器总线时钟使能

  GPIO_PF4AFSEL_PF4AFS_Set(GPIO_PF4AFSEL_PF4AFS_LPUART_TX);//PF4选择LPUART_TX
  //LPUART IO 配置
  AltFunIO(GPIOF, GPIO_Pin_3, 2);   //PF3 LPUART RX 除匹配接收 假如外部没有上拉电阻建议配置上拉电阻
  AltFunIO(GPIOF, GPIO_Pin_4, 0);   //PF4 LPUART TX

  /*NVIC中断配置*/
  NVIC_DisableIRQ(LPUART_IRQn);
//NVIC_SetPriority(LPUART_IRQn,2);//中断优先级配置
//NVIC_EnableIRQ(LPUART_IRQn);


  //UART初始化配置
  init_para.BaudRate = 9600;      //波特率
  init_para.DataBit = Eight8Bit;    //数据位数
  init_para.ParityBit = EVEN;     //奇偶校验
  init_para.StopBit = OneBit;     //停止位

  LPUART_SInit(&init_para); //初始化uart
}
```

图 2-2 LPUART_Init 函数

5、配置 LPUEN 寄存器打开接收和发送使能

配置 LPUEN 寄存器比较特殊，接收和发送使能使用的是异步逻辑，所以写完寄存器后，寄存器值不会立刻变化。在写完寄存器后一定要判断寄存器值是否真的已经写入。

举例接收使能函数如图 2-3 所示

```
/*********************************
LPUART  接收使能设置函数
输入:   0: 关闭接收
        1: 打开接收
输出:   无
*********************************/
void LPUART_LPUEN_RXEN_Setable(FunState NewState)
{
  if (NewState == ENABLE)
  {
    LPUART->LPUEN |= (LPUART_LPUEN_RXEN_Msk);
    while(((LPUART->LPUEN)&LPUART_LPUEN_RXEN_Msk)==0);
  }
  else
  {
    LPUART->LPUEN &= ~(LPUART_LPUEN_RXEN_Msk);
    while(((LPUART->LPUEN)&LPUART_LPUEN_RXEN_Msk)!=0);
  }
```

图 2-3 LPUART 接收使能函数

6　发送和接收数据寄存器

发送数据时，在写发送数据寄存器前一定要判断 buffer 是否为空，可以判断 LPUSTA 的 bit6 TXE，或其他发送相关标志位，但不建议使用 TC，TC 的值不能同步的反应发送寄存器的实时变化。TC_IF 和 TXIF 需要软件写 1 清 0，TXE 和 TC 硬件自动清 0。在低功耗应用中，程序在写完发送数据寄存器即可执行休眠，不必要等发送完成后再休眠。

在接收时 RXIF、RXNEGIF、START、MATCH 标志位都是软件写 1 清 0。

注: LPUART 模块只有 RXEN、TXEN、TC 存在同步问题，其余寄存器都是实时变化的。

2.2　LPUART 硬件设计

本章需要用到的硬件如图 2-4 所示

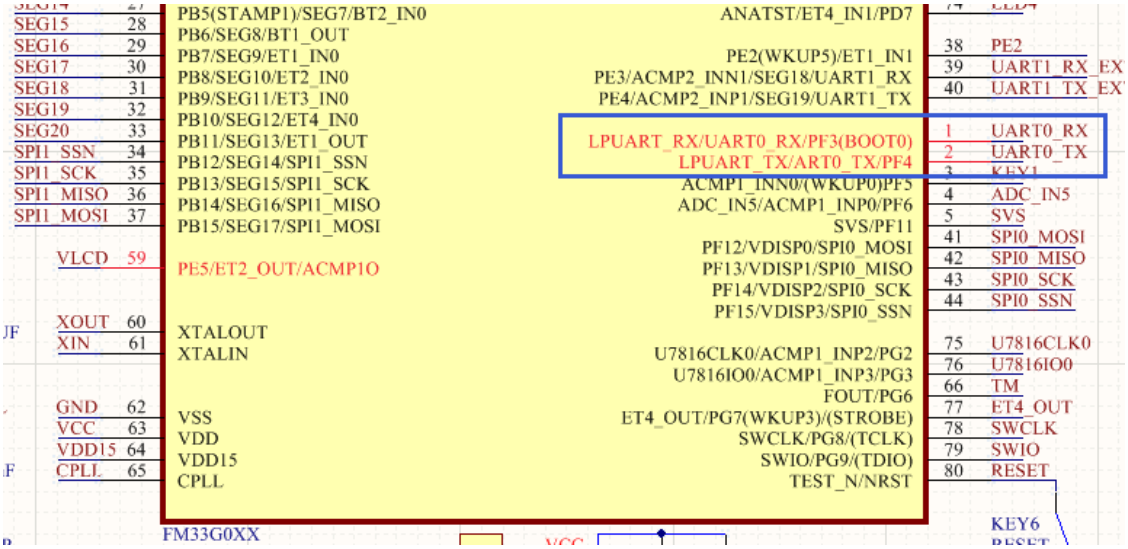


图 2-4 硬件原理图设计

对应的 PCB 位置如图 2-5 所示

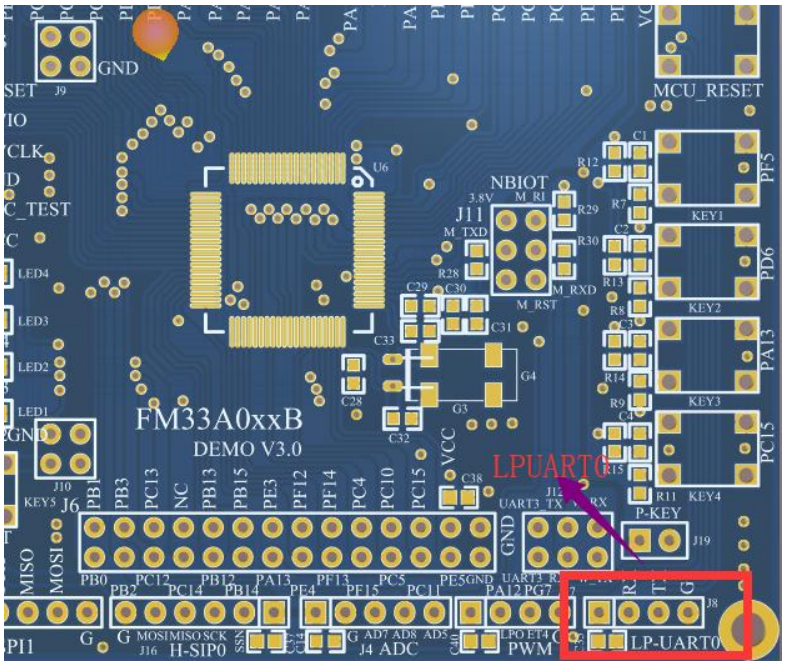


图 2-5 硬件 PCB 版图

3 参考用例

参考例程如图 3-1 所示

图 3-1 LUART 参考示例

3.1 LPUART 查询收发例程

如图 3-2 所示

```
void Test_LPUart(void)
{
  uint08 TestTxData[12] = {0x61,0x69,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x10,0x11,0x12};
  uint08 i;
  uint08 tmp08;

  LPUART_LPUEN_RXEN_Setable(ENABLE);    //打开接收使能
  LPUART_LPUEN_TXEN_Setable(ENABLE);    //打开发送使能

  //清除发送中断标志
  LPUART_LPUIF_TXIF_Clr();
  LPUART_LPUIF_TC_IF_Clr();

  //查询发送数组
  for(i=0; i<12; i++)
  {
    LPUART_LPUTXD_Write(TestTxData[i]);   //将发送数据写入发送寄存器
    while(0 == LPUART_LPUIF_TC_IF_Chk()); //等待发送完成
    LPUART_LPUIF_TXIF_Clr(); //清除发送中断标志
    LPUART_LPUIF_TC_IF_Clr(); //清除发送中断标志
  }

  //转发接收到的数据
  while(1)
  {
    IWDT_Clr();              //清系统看门狗
    if(SET == LPUART_LPUIF_RXIF_Chk())    //等待接收到一个字节
    {
      tmp08 = LPUART_LPURXD_Read();   //uart接收数据
      LPUART_LPUIF_RXNEGIF_Clr();     //清除接收中断标志
      LPUART_LPUIF_RXIF_Clr();        //清除接收中断标志
      LPUART_LPUTXD_Write(tmp08);     //将收到的数据发送回去
    }
  }
}
```

图 3-2 LUART 查询例程

程序解释：

程序先发一组数据，然后处于等待状态，将接收到的数据回发

3.2 LPUART 接收数据匹配唤醒例程

如图 3-3 所示

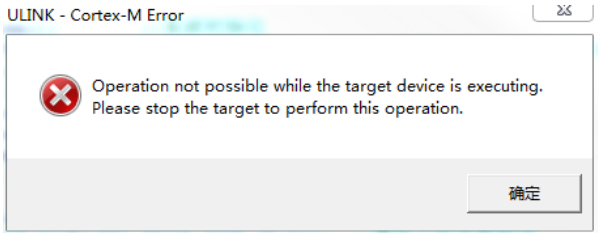
```
void LPUart_Init(void)
{
  LPUART_SInitTypeDef init_para;

  RCC_PERCLK_SetableEx(LPUFCKEN, ENABLE); //LPUART功能时钟使能
  RCC_PERCLK_SetableEx(LPUARTCKEN, ENABLE); //LPUART寄存器总线时钟使能

  GPIO_PF4AFSEL_PF4AFS_Set(GPIO_PF4AFSEL_PF4AFS_LPUART_TX);//PF4选择LPUART_TX
  //LPUART IO 配置
  AltFunIO(GPIOF, GPIO_Pin_3, 2);   //PF3 LPUART RX
  AltFunIO(GPIOF, GPIO_Pin_4, 0);   //PF4 LPUART TX

  /*NVIC中断配置*/
  NVIC_DisableIRQ(LPUART_IRQn);
  NVIC_SetPriority(LPUART_IRQn,2);//中断优先级配置
  NVIC_EnableIRQ(LPUART_IRQn);

  //UART初始化配置
  init_para.BaudRate = 9600;      //波特率
  init_para.DataBit = Eight8Bit;    //数据位数
  init_para.ParityBit = EVEN;     //奇偶校验
  init_para.StopBit = OneBit;     //停止位

  LPUART_SInit(&init_para); //初始化uart(接收中断使用的是接收数据匹配)
}

void Test_LPUart(void)
{
  LPUART_LPUEN_RXEN_Setable(ENABLE);    //打开接收使能
  while(((LPUART->LPUEN)&0x01)==0);     //寄存器与模块不同步，必须确认已经写进寄存器，防止写其他bit时覆盖掉

  //清除接收中断标志
  LPUART_LPUIF_RXNEGIF_Clr();     //清除接收中断标志
  LPUART_LPUIF_RXIF_Clr();        //清除接收中断标志
  LPUART_LPUSTA_MATCH_Clr();      //清除数据匹配标志
  LPUART_LPUSTA_START_Clr();      //清除起始位标志

  LPUART_COMPARE_Write(0x33);//写数据匹配寄存器
  CloseeIO(GPIOF, GPIO_Pin_4);    //PF4 开漏输出高
  LPUART_LPUCON_RXIE_Setable(ENABLE);//开接收中断使能
}
```

图 3-3 LUART 接收匹配唤醒例程

程序解释：

LPUART 处于睡眠模式下 RXD 引脚一直处于监听状态，直到特定事件（串口助手发送 0X33）到来后唤醒芯片退出休眠模式，执行 main 函数中 for 循环代码

注：休眠模式下，仿真状态下不能打断点，否则会提示



3.3 LPUART 中断收发例程

如图 3-4 所示

```
void Test_LPUart(void)
{
  LPUART_LPUEN_RXEN_Setable(ENABLE);     //打开接收使能
  LPUART_LPUEN_TXEN_Setable(ENABLE);     //打开发送使能

  //中断发送数组
  LPUARTOp.TxBuf = TestTxData;
  LPUARTOp.TxLen = 12;
  LPUARTOp.TxOpc = 0 + 1;

  //清除发送中断标志
  LPUART_LPUIF_TXIF_Clr();
  LPUART_LPUIF_TC_IF_Clr();

  LPUART_LPUCON_TCIE_Setable(ENABLE); //使能发送中断
  LPUART_LPUTXD_Write(LPUARTOp.TxBuf[0]);//启动第一个数据的发送
  TicksDelayMs( 50, NULL );//软件延时

  LPUART_LPUCON_TCIE_Setable(DISABLE); //禁止发送中断
  LPUART_COMPARE_Write(0x33);//写数据匹配寄存器
  LPUART_LPUCON_RXIE_Setable(ENABLE);//使能接收中断

}

void LPUART_IRQHandler(void)
{
  uint08 tmp08;

  //接收中断处理
  if((ENABLE == LPUART_LPUCON_RXIE_Getable())
    &&(SET == LPUART_LPUSTA_MATCH_Chk()))//接收数据匹配中断
  {
    tmp08 = LPUART_LPURXD_Read();
  //  LPUART_LPUIF_RXNEGIF_Clr();      //清除接收中断标志
  //  LPUART_LPUIF_RXIF_Clr();         //清除接收中断标志
    LPUART_LPUSTA_MATCH_Clr();       //清除数据匹配标志
  //  LPUART_LPUSTA_START_Clr();       //清除起始位标志
  }

  //发送中断处理
  if((ENABLE == LPUART_LPUCON_TCIE_Getable())
    &&(SET == LPUART_LPUIF_TC_IF_Chk()))
  {
    //发送指定长度的数据
    if(LPUARTOp.TxOpc < LPUARTOp.TxLen)
    {
      LPUART_LPUTXD_Write(LPUARTOp.TxBuf[LPUARTOp.TxOpc]); //发送一个数据
      LPUARTOp.TxOpc++;
    }
    //清除发送中断标志
   // LPUART_LPUIF_TXIF_Clr();
    LPUART_LPUIF_TC_IF_Clr();
  }
}
```

图 3-4 LUART 中断收发例程

程序解释：

1、程序运行后一直处于 main 函数的 for 循环中，例程 Test_LPUart（）函数中，发送完第一组数据后关闭了发送中断，用户根据自己实际情况修改即可。

2、当串口助手发送 0x33 时，接受中断执行，例程中中断未执行任何操作，仅仅消除数据匹配标志，用户根据自己实际情况添加操作即可。

3.4 普通 UART 模拟 LPUART

3.4.1 原理

将普通 UART 的 RX 和 WKUP 相连接，在芯片处于 sleep 或 deepsleep 时，当外部发送串口数据时，起始位通过 WKUP 唤醒芯片，然后再接收串口的数据。

3.4.2 使用说明

制约串口接收数据的主要因素为唤醒芯片的时间，FM33G0XX 系列芯片的 WKUP 的中断源有两个：一个为不可屏蔽中断 NMI，一个为#38 中断，可以通过寄存器 PINWKEN 来配置。当芯片处于 sleep 或 deepsleep 使用 NMI 中断唤醒时间约为 7.9us，

3.4.3 硬件设计

本章需要用到的硬件如图 3-5 所示

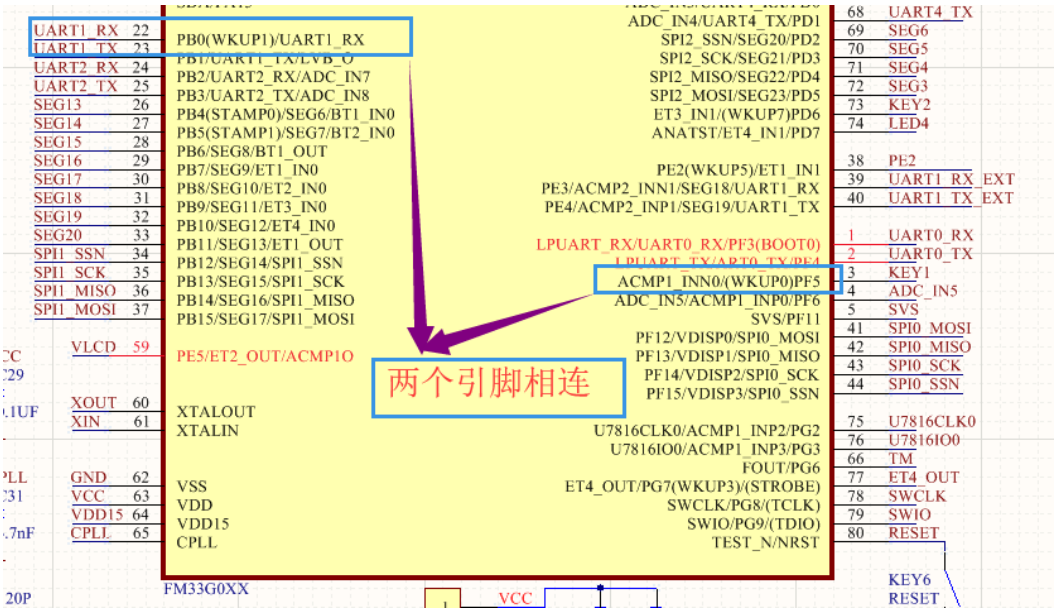


图 3-5 UART 模拟 LUART 硬件设计

对应的 PCB 位置如图 3-6 所示

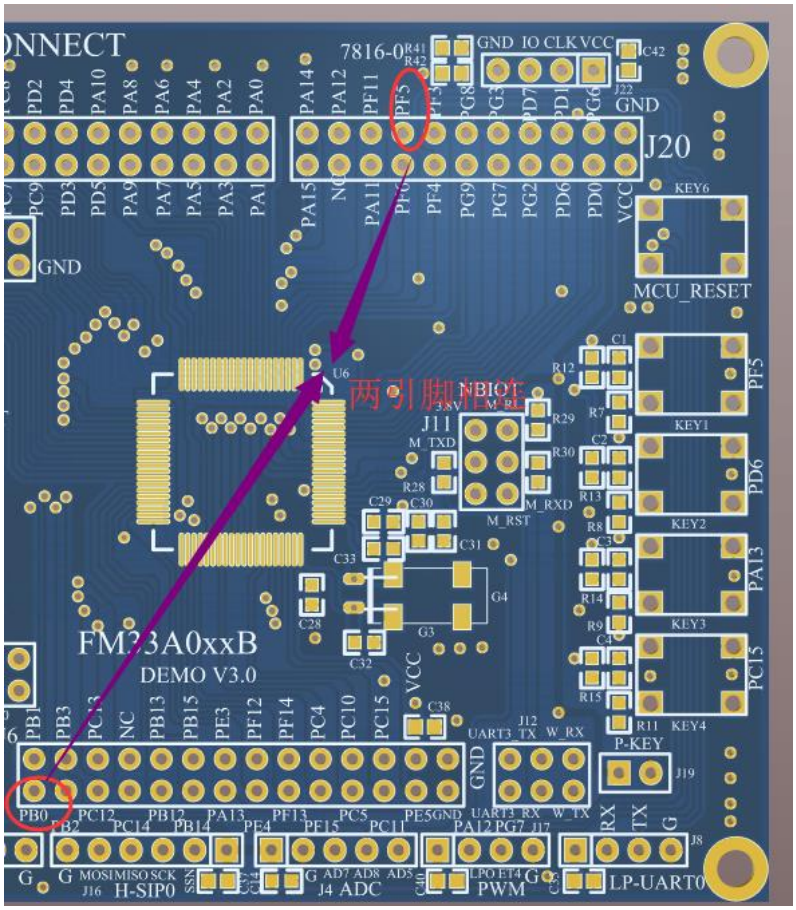


图 3-5 UART 模拟 LUART-PCB 版图

3.4.3 软件实现

1、WKUP 配置

```
void Wakeup_Cfg(void)
{

  InputtIO( GPIOF, GPIO_Pin_5, IN_PULLUP ); //PF5;//DISPKEY,带上拉输入
  GPIO_EXTI_Init(GPIOF, GPIO_Pin_5, EXTI_DISABLE);//关闭IO中断

  NVIC_DisableIRQ(GPIO_IRQn);
  NVIC_ClearPendingIRQ(GPIO_IRQn);
  //NWKUP连接到了cpu的NMI不可屏蔽中断，不受NVIC控制，不受全局中断使能控制，唤醒后必然进
  GPIO_PINWKEN_SetableEx(PINWKEN_PF5, ENABLE);  //使能PF5的NWKUP1功能
}
```

2、配置 UART

```
void Test_Uartx(UARTx_Type* UARTx)
{
  Uartx_Init(UARTx);//初始化uart配置

  UARTx_RXSTA_RXEN_Setable(UARTx, ENABLE);     //打开接收使能
  UARTx_TXSTA_TXEN_Setable(UARTx, ENABLE);     //打开发送使能

  UART_UARTIF_RxTxIF_ClrEx(UARTx);  //清除发送中断标志

  UART_UARTIE_RxTxIE_SetableEx(UARTx, RxInt, ENABLE);//打开接收中断

}
```

程序解释：如图 3-6 所示

```
189 }
190 int main (void)
191 {
192   Init_System();         //系统初始化
193   IWDT_IWDTCFG_IWDTSLP4096S_Setable(ENABLE);  //配置休眠时是否启用4096s长周期
194   __disable_irq();       //关闭全局中断
195   RTC_Cfg();
196   Wakeup_Cfg(); //唤醒源配置
197   Test_Uartx(UART1);//UART测试
198   Test_Sleep(); //休眠测试
199
200     for( ; ; )
201   {
202     IWDT_Clr();
203     RTC_RTCIF_ClrEx(RTC_RTCIE_HOUR_IE_Msk); //清除小时中断标志
204     NVIC_ClearPendingIRQ(RTC_IRQn); //清挂起标志
205     NVIC_ClearPendingIRQ(UART1_IRQn); //清挂起标志
206       if(NwkupFlag == 0x55)
207       {
208         NwkupFlag = 0;
209         break;
210       }
211   }
212 }
```

说明：当芯片处于休眠状态时，由于 WKUP0 接到 UART1_RX 引脚上，通过串口向 UART1 发送数据，起始位下降沿触发 WKUP0 而唤醒芯片。

现象：当通过串口发送任意数据时，处于休眠状态的芯片可运行到断点处。

版本信息

版本号	发布日期	更改说明
1.0	2018.9	首次发布

附录

芯片寄存器介绍

地址	名称	符号
0x40010400	接收数据寄存器	LPURXD
0x40010404	发送数据寄存器	LPUTXD
0x40010408	状态寄存器	LPUSTA
0x4001040C	控制寄存器	LPUCON
0x40010410	中断标志寄存器	LPUIF
0x40010414	波特率寄存器	LPUBAUD
0x40010418	接收使能寄存器	LPUEN
0x4001041C	数据匹配寄存器	COMPARE
0x40010420	波特率调制控制寄存器	MODU

接收数据寄存器

名称	LPURXD							
地址	0x40010400							
位	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
位名	-							
位权限	U-0							
位	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
位名	-							
位权限	U-0							
位	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
位名	-							
位权限	U-0							
位	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位名	DATA							
位权限	R-0							

位号	位名	说明
31:8	--	**未实现：读为0**
7:0	**LPURXD**	接收数据缓冲

发送数据寄存器

名称	LPUTXD							
地址	0x40010404							
位	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
位名	-							
位权限	U-0							
位	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
位名	-							
位权限	U-0							
位	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
位名	-							
位权限	U-0							
位	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位名	DATA							
位权限	R-0							

位号	位名	说明
31:8	--	**未实现：读为0**
7:0	**LPUTXD**	发送数据缓冲

状态标志寄存器

名称	LPUSTA							
地址	0x40010408							
位	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
位名	-							
位权限	U-0							
位	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
位名	-							
位权限	U-0							
位	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
位名	-							
位权限	U-0							
位	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位名	TC	TXE	START	PERR	FERR	RXOV	RXF	MATCH
位权限	R/Dy-1	R/Dy-1	R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0	R	R/W1C-0

位号	位名	说明
31:8	--	**未实现：读为0**
7	**TC**	发送完成标志，当一帧数据发送完成且发送 buffer 为空时置位。 数据发送时清零。
6	**TXE**	发送 buffer 空标志 硬件置位，软件向发送 buffer 写数据时自动清零
5	**START**	起始位检测标志，写 1 清零
4	**PERR**	校验位错误，写 1 清零
3	**FERR**	帧格式错误，写 1 清零
2	**RXOV**	接收缓冲溢出，写 1 清零
1	**RXF**	接收缓冲满，读 LPUDATA 寄存器清零
0	**MATCH**	数据匹配标志，表示接收缓冲区内的数据与比较寄存器相同，写 1 清零

控制寄存器

名称	LPUCON							
地址	0x4001040C							
位	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
位名	-							
位权限	U-0							
位	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
位名	-							
位权限	U-0							
位	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
位名	-			TXPOL	TCIE	TXIE	NEDET	PAREN
位权限	U-0			R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
位	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位名	PTYP	SL	DL	RXPOL	ERRIE	RXIE	RXEV	
位权限	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-00	

位号	位名	说明
31:13	--	**未实现：读为0**
12	**TXPOL**	数据发送极性 0：非取反 1：取反

位号	位名	说明
11	**TCIE**	发送完成中断使能 0：禁止发送完成中断 1：允许发送完成中断
10	**TXIE**	发送 buffer 空中断使能 0：禁止发送 buffer 空中断 1：允许发送 buffer 空中断
9	**NEDET**	下降沿采样使能位 0：不使用 32k 时钟下降沿检测 start bit 1：使用 32k 时钟下降沿检测 start bit
8	**PAREN**	校验位使能 0：数据帧无奇偶校验位 1：数据帧有奇偶校验位
7	**PTYP**	校验位类型 0：偶校验 1：奇校验
6	**SL**	停止位长度 0：1bit 1：2bits
5	**DL**	数据长度 0：8bits 1：7bits
4	**RXPOL**	接收极性 0：非取反 1：取反
3	**ERRIE**	错误中断使能 0：禁止接收错误中断 1：允许接收错误中断
2	**RXIE**	接收中断使能 0：禁止接收中断 1：允许接收中断

位号	位名	说明
1:0	**RXEV**	接收中断事件配置，用于控制何种事件下向 CPU 提供接收中断 00：START 位检测唤醒 01：1byte 数据接收完成 10：接收数据匹配成功 11：RXD 下降沿唤醒

中断标志寄存器

名称	LPUIF							
地址	0x40010410							
位	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
位名	-							
位权限	U-0							
位	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
位名	-							
位权限	U-0							
位	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
位名	-							
位权限	U-0							
位	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位名	-				TC_IF	TXIF	RXNEGIF	RXIF
位权限	U-0				R/W1C-0	R/W1C-0	R/W1C-0	R/W1C-0

位号	位名	说明
31:4	**--**	**未实现：读为0**
3	**TC_IF**	发送完成中断标志 1：发送完一帧数据后中断产生 0：无中断产生
2	**TXIF**	发送 buffer 空中断标志 1：发送 buffer 空后中断产生 0：无中断产生
1	**RXNEGIF**	RXD 下降沿中断标志 1：中断产生 0：无中断产生

位号	位名	说明
0	**RXIF**	接收完成中断标志 1：接收完一帧数据后中断产生 0：无中断产生

波特率寄存器

名称	**LPUBAUD**							
地址	0x40010414							
位	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
位名	-							
位权限	U-0							
位	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
位名	-							
位权限	U-0							
位	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
位名	-							
位权限	U-0							
位	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位名	-					BAUD		
位权限	U-0					R/W-000		

位号	位名	说明
31:3	--	**未实现：读为0**
2:0	**BAUD**	波特率控制（bps） 000：9600 001：4800 010：2400 011：1200 100：600 101/110/111：300

发送接收使能寄存器

名称	LPUEN							
地址	0x40010418							
位	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
位名	-							
位权限	U-0							
位	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
位名	-							
位权限	U-0							
位	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
位名	-							
位权限	U-0							
位	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位名	-						TXEN	RXEN
位权限	U-0						R/W-0	R/W-0

位号	位名	说明
31:2	--	**未实现：读为0**
1	**TXEN**	发送使能 0：关闭 LPUART 发送 1：打开 LPUART 发送
0	**RXEN**	接收使能 0：关闭 LPUART 接收 1：打开 LPUART 接收

数据匹配寄存器

名称	COMPARE							
地址	0x4001041C							
位	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
位名	-							
位权限	U-0							
位	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
位名	-							
位权限	U-0							
位	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
位名	-							

位权限	U-0							
位	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位名	COMPARE							
位权限	R/W-0							

位号	位名	说明
31:8	--	**未实现：读为0**
7:0	**COMPARE**	比较数据，如果 RXEV=10/11，当接收缓冲区内的数据与 COMPARE 相同时，触发接收完成中断

调制控制寄存器

名称	**MCTL**							
地址	0x40010420							
位	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
位名	-							
位权限	U-0							
位	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
位名	-							
位权限	U-0							
位	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
位名	-				MCTL			
位权限	U-0				R/W-0000			
位	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
位名	MCTL							
位权限	R/W-00000000							

位号	位名	说明
31:12	--	**未实现：读为0**
11:0	**MCTL**	LPUART 每个 bit 的调制控制信号

上海复旦微电子集团股份有限公司销售及服务网点

上海复旦微电子集团股份有限公司
地址：上海市国泰路 127 号 4 号楼
邮编：200433
电话：(86-021) 6565 5050
传真：(86-021) 6565 9115

上海复旦微电子（香港）股份有限公司
地址：香港九龙尖沙咀东嘉连威老道 98 号东海商业中心 5 楼 506 室
电话：(852) 2116 3288　2116 3338
传真：(852) 2116 0882

北京办事处
地址：北京市东城区东直门北小街青龙胡同 1 号歌华大厦 B 座 423 室
邮编：100007
电话：（86-10）8418 6608
传真：（86-10）8418 6211

深圳办事处
地址：深圳市华强北路 4002 号圣廷苑酒店世纪楼 1301 室
邮编：518028
电话：(86-0755) 8335 0911　8335 1011　8335 2011 8335 0611
传真：(86-0755) 8335 9011

台湾办事处
地址：台北市 114 内湖区内湖路一段 252 号 12 楼 1225 室
电话：(886-2) 7721 1889
传真：(886-2) 7722 3888

新加坡办事处
地址：237, Alexandra Road, #07-01, The Alexcier, Singapore 159929
电话：(65) 6472 3688
传真：(65) 6472 3669

北美办事处
地址：2490 W. Ray Road Suite#2 Chandler, AZ 85224 USA
电话：(480) 857-6500 ext 18
公司网址：http://www.fmsh.com/