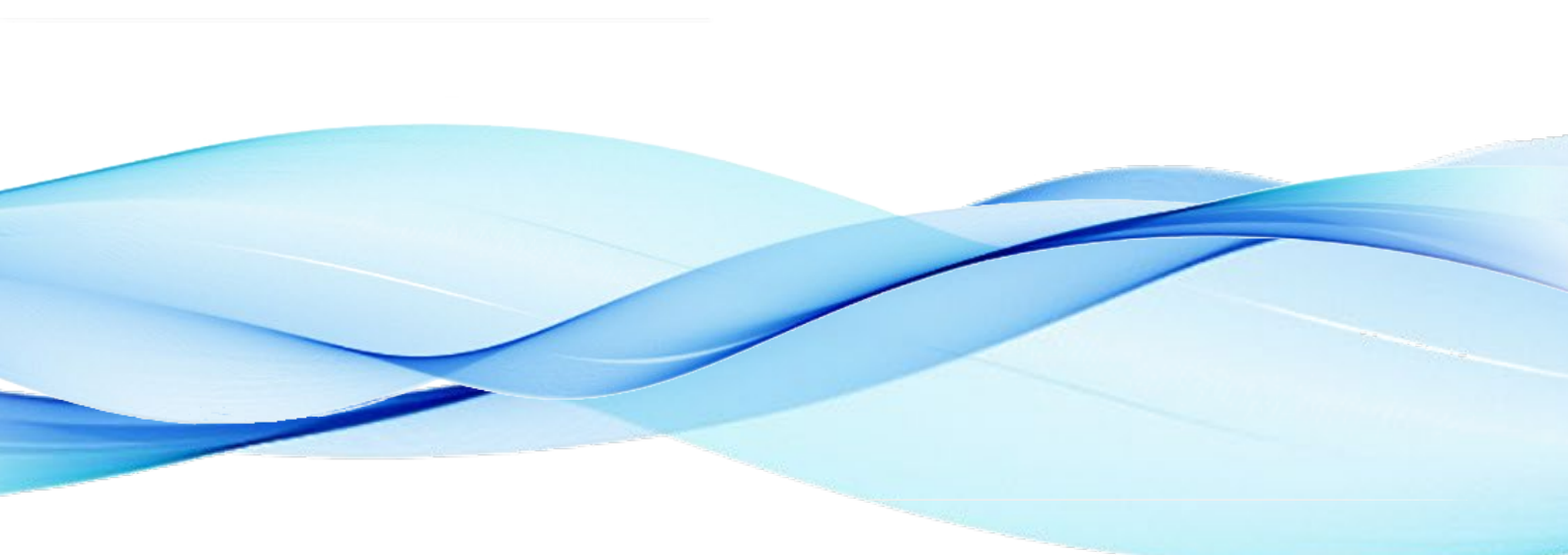




Si24R1_AvrMega8

Demo 程序说明



官网: www.ashining.com

邮箱: support@ashining.com

地址: 四川省·成都市·高新西区百草路898号
智能信息产业园2层、5层

Si24R1_AvrMega8 的 demo 程序说明

本 demo 程序是基于 AvrMega8 单片机和 Si24R1 开发设计的。本程序包含了主函数文件 main.c, SPI 文件 drv_spi.c, 串口文件 drv_uart.c, 指示灯相关函数的文件 drv_led.c, 按键相关函数文件 drv_button.c, 延时函数文件 drv_delay.c 以及 Si24R1 的驱动文件 drv_RF24L01.c 等。

本程序实现的功能是使用 Si24R1 进行透明传输的功能, 但只支持单独的接收或单独的发送。若想实现收发一体的功能需要用户自行修改程序。发送功能中分为了 2 种模式, 固定发送模式和自由发送模式, 由按键控制。自由发送模式是发送串口收到的数据。

1. 切换发送功能或接收功能的进行编译

打开工程文件后, 在 main.c 文件中有 __RF24L01_TX_TEST__ 的一个宏定义, 若该参数未被定义的话发送功能则未被编译, 若就当前状态进行编译下载, 则该模块有了接收功能。要想编译下载发送功能的程序, 需要点开 main.h 文件, 将 #define __RF24L01_TX_TEST__ 释放出来即可。如图:

打开 main.h 文件

```
/**
 * @author 泽耀科技 ASHINING
 * @version V3.0
 * @date 2016-10-08
 * @brief 主配置H文件
 * @attention
 *
 * 官网 : http://www.ashining.com
 * 淘宝 : https://shop105912646.taobao.com
 * 阿里巴巴: https://cdzeyao.1688.com
 */

#ifndef __MAIN_H__
#define __MAIN_H__

#include "drv_RF24L01.h"
#include "drv_uart.h"
#include "drv_button.h"
#include "drv_delay.h"
#include "drv_led.h"

// #define __RF24L01_TX_TEST__
// #define __USE_SOFT_SPI_INTERFACE__

/** **@@ 如果测试发送功能则需
 **@@ 如果使用软件SPI则需
```

释放掉圈出部分

```
/**
*****
* @author 泽耀科技 ASHINING
* @version V3.0
* @date 2016-10-08
* @brief 主配置H文件
*****
* @attention
*
* 官网 : http://www.ashining.com
* 淘宝 : https://shop105912646.taobao.com
* 阿里巴巴: https://cdzeyao.1688.com
*****
*/

#ifndef __MAIN_H__
#define __MAIN_H__

#include "drv_RF24L01.h"
#include "drv_uart.h"
#include "drv_button.h"
#include "drv_delay.h"
#include "drv_led.h"

#define __RF24L01_TX_TEST__ //***@@ 如果测试发送功能则需要定义该
// #define __USE_SOFT_SPI_INTERFACE__ //***@@ 如果使用软件SPI则需要定义该
```

现在的主程序，发送功能可编译，接受功能部分不可编译

2. 更改串口波特率

本程序默认的串口波特率是 9600，我们可以通过更改 drv_uart_init() 中 UBRR1 的值来更改串口波特率。drv_uart_init() 在 drv_uart.c 文件中如图：

```
const char *g_Ashining = "ashining";
uint8_t g_TxMode = 0, g_UartRxFlag = 0;
uint8_t g_UartRxBuffer[ 100 ] = { 0 };
uint8_t g_RF24L01RxBuffer[ 32 ] = { 0 };
uint8_t g_test[ 8 ] = { 1, 2, 3, 4, 5, 6, 7, 8 };

/**
* @brief : 主函数
* @param : 无
* @note : 无
* @retval: 无
*/
int main( void )
{
    uint8_t i = 0;

    //串口初始化
    drv_uart_init();

    //LED初始化
    drv_led_init();

    //SPI初始化
    drv_spi_init();

    //RF24L01引脚初始化
    NRF24L01_Gpio_Init();

    //检测nRF24L01
    NRF24L01_check();
    RF24L01_Init();
```

```
*/
void drv_uart_init( )
{
    /** 串口引脚初始化 */
    //TX 配置为输出 RX 配置为输入
    UART_TX_GPIO_ODR |= 0x01 << UART_TX_GPIO_BIT;
    UART_RX_GPIO_ODR &= (uint8_t)( ~( 0x01 << UART_RX_GPIO_BIT ));

    UART_TX_GPIO_PORT |= 0x01 << UART_TX_GPIO_BIT; //TX 引脚默认置高
    UART_RX_GPIO_PORT |= 0x01 << UART_RX_GPIO_BIT; //RX 引脚上拉

    /** 串口外设初始化 */

    UCSRB = 0x00;
    UCSRB |= ( 0x01 << RXEN ) | ( 0x01 << TXEN ); //使能接收 发送

    UCSRC |= ( 0x01 << UCSZ1 ) | ( 0x01 << UCSZ0 ); //8位数据位 1个停止位 无校验 异步通信

    UCSRC &= (uint8_t)( ~( 0x01 << URSEL )); //更新UBRRH
    UBRRH &= (uint8_t)( ~0x0F );
    UBRRL = 49; //波特率9600 8MHz晶振是理论应该为51 实际调试因为RC误差改为49
}

/==
```

本程序默认波特率为 9600 则设置 UBRRL 为 49。例如若要设置波特率为 4800 则理论应设 UBRRL 为 103。

3. 更改 Si24R1 的通信地址

若需更改 Si24R1 的通信地址，我们需要打开 drv_RF24L01.h 文件，更改 INIT_ADDR 的宏定义参数。值得注意的是发送模块和接收模块的通信地址要一致才能通信。

```
/**
*****
* @author 泽耀科技 ASHINING
* @version V3.0
* @date 2016-10-08
* @brief NRF24L01配置H文件
*****
* @attention
*
* 官网 : http://www.ashining.com
* 淘宝 : https://shop105912646.taobao.com
* 阿里巴巴: https://cdzeyao.1688.com
*****
*/

#ifndef __DRV_RF24L01_H__
#define __DRV_RF24L01_H__

#include "drv_spi.h"

/** 配置和选项定义 */
#define DYNAMIC_PACKET 1 //1:动态数据包, 0:固定
#define FIXED_PACKET_LEN 32 //包长度
#define REPEAT_CNT 15 //重复次数
#define INIT_ADDR 0x34, 0x43, 0x10, 0x10, 0x01
/** RF24L01硬件接口定义 */
```

4. 更改 Si24R1 的通信配置

若需更改 Si24R1 的通信配置则需要在 drv_rf24l1.c 文件中的 RF24L01_Init()函数修改对应参数。其中通信配置中最重要的是信道，空速，发射功率等。需注意的是发送方与接收方的信道，空速，发射功率都需一致。

a. 若需要更改信道则更改 RF_CH 寄存器的参数。

```
void RF24L01_Init( void )
{
    uint8_t addr[5] = {INIT_ADDR};

    RF24L01_SET_CE_HIGH();
    NRF24L01_Clear_IRQ_Flag( IRQ_ALL );
    #if DYNAMIC_PACKET == 1

        NRF24L01_Write_Reg( DYNPD, ( 1 << 0 ) ); //使能通道1动态数据长度
        NRF24L01_Write_Reg( FEATRUE, 0x07 ); //设置特征寄存器，使能动态负载长度，使能ACK负载，使
        NRF24L01_Read_Reg( DYNPD ); //读取使能动态负载长度
        NRF24L01_Read_Reg( FEATRUE ); //读取特征寄存器

    #elif DYNAMIC_PACKET == 0

        L01_WriteSingleReg( L01REG_RX_PW_P0, FIXED_PACKET_LEN ); //固定数据长度

    #endif //DYNAMIC_PACKET

    NRF24L01_Write_Reg( CONFIG, /*( 1<<MASK_RX_DR ) | */ //接收中断
        ( 1 << EN_CRC ) | //使能CRC 1个字节
        ( 1 << PWR_UP ) ); //开启设备
    NRF24L01_Write_Reg( EN_AA, ( 1 << ENAA_P0 ) ); //通道0自动应答
    NRF24L01_Write_Reg( EN_RXADDR, ( 1 << ERX_P0 ) ); //通道0接收
    NRF24L01_Write_Reg( SETUP_AW, AW_5BYTES ); //地址宽度 5个字节
    NRF24L01_Write_Reg( SETUP_RETR, ARD_4000US | //重复等待时间 250us
        ( REPEAT_CNT & 0x0F ) ); //初始化通道
    NRF24L01_Write_Reg( RF_CH, 60 ); //设置通信速度
    NRF24L01_Write_Reg( RF_SETUP, 0x26 );

    NRF24L01_Set_TxAddr( &addr[0], 5 ); //设置TX地址
    NRF24L01_Set_RxAddr( 0, &addr[0], 5 ); //设置RX地址
}
```

寄存器参数设置：(2.4GHz---2.525GHz)

05	RF_CH				RF Channel
	Reserved	7	0	R/W	Only '0' allowed
	RF_CH	6:0	0000010	R/W	Sets the frequency channel nRF24L01+ operates on

b. 若需更改空速和发射功率则更改 RF_SETUP 寄存器的参数。

```
/**
 * @brief :RF24L01模块初始化
 * @param :无
 * @note :无
 * @retval:无
 */
void RF24L01_Init( void )
{
    uint8_t addr[5] = {INIT_ADDR};

    RF24L01_SET_CE_HIGH();
    NRF24L01_Clear_IRQ_Flag( IRQ_ALL );
    #if DYNAMIC_PACKET == 1

        NRF24L01_Write_Reg( DYNPD, ( 1 << 0 ) ); //使能通道1动态数据长度
        NRF24L01_Write_Reg( FEATRUE, 0x07 ); //设置特征寄存器, 使能动态负载长度, 使能ACK负载,
        NRF24L01_Read_Reg( DYNPD ); //读取使能动态负载长度
        NRF24L01_Read_Reg( FEATRUE ); //读取特征寄存器

    #elif DYNAMIC_PACKET == 0

        L01_WriteSingleReg( L01REG_RX_PW_P0, FIXED_PACKET_LEN ); //固定数据长度
    #endif //DYNAMIC_PACKET

    NRF24L01_Write_Reg( CONFIG, /*( 1 << MASK_RX_DR ) | */ //接收中断
                        ( 1 << EN_CRC ) | //使能CRC 1个字节
                        ( 1 << PWR_UP ) ); //开启设备
    NRF24L01_Write_Reg( EN_AA, ( 1 << ENAA_P0 ) ); //通道0自动应答
    NRF24L01_Write_Reg( EN_RXADDR, ( 1 << ERX_P0 ) ); //通道0接收
    NRF24L01_Write_Reg( SETUP_AW, AW_5BYTES ); //地址宽度 5个字节
    NRF24L01_Write_Reg( SETUP_RETR, ARD_4000US | //重复等待时间 250us
                        ( REPEAT_CNT & 0x0F ) );
    NRF24L01_Write_Reg( RF_CH, 60 ); //初始化通道
    NRF24L01_Write_Reg( RF_SETUP, 0x26 ); //设置通信速度

    NRF24L01_Set_TxAddr( &addr[0], 5 ); //设置TX地址
    NRF24L01_Set_RxAddr( 0, &addr[0], 5 ); //设置RX地址
}
```

寄存器参数设置：

06	RF_SETUP				RF Setup Register
	CONT_WAVE	7	0	R/W	Enables continuous carrier transmit when high.
	Reserved	6	0	R/W	Only '0' allowed
	RF_DR_LOW	5	0	R/W	Set RF Data Rate to 250kbps. See RF_DR_HIGH for encoding.
	PLL_LOCK	4	0	R/W	Force PLL lock signal. Only used in test
	RF_DR_HIGH	3	1	R/W	Select between the high speed data rates. This bit is don't care if RF_DR_LOW is set. Encoding: [RF_DR_LOW, RF_DR_HIGH]: '00' - 1Mbps '01' - 2Mbps '10' - 250kbps '11' - Reserved
	RF_PWR	2:1	11	R/W	Set RF output power in TX mode '00' - -18dBm '01' - -12dBm '10' - -6dBm '11' - 0dBm
	Obsolete	0			Don't care

例如参数为 0x26:空速为 250kbps, 发射功率为 0dBm。