



nRF24L01_HC32L110C4UA

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

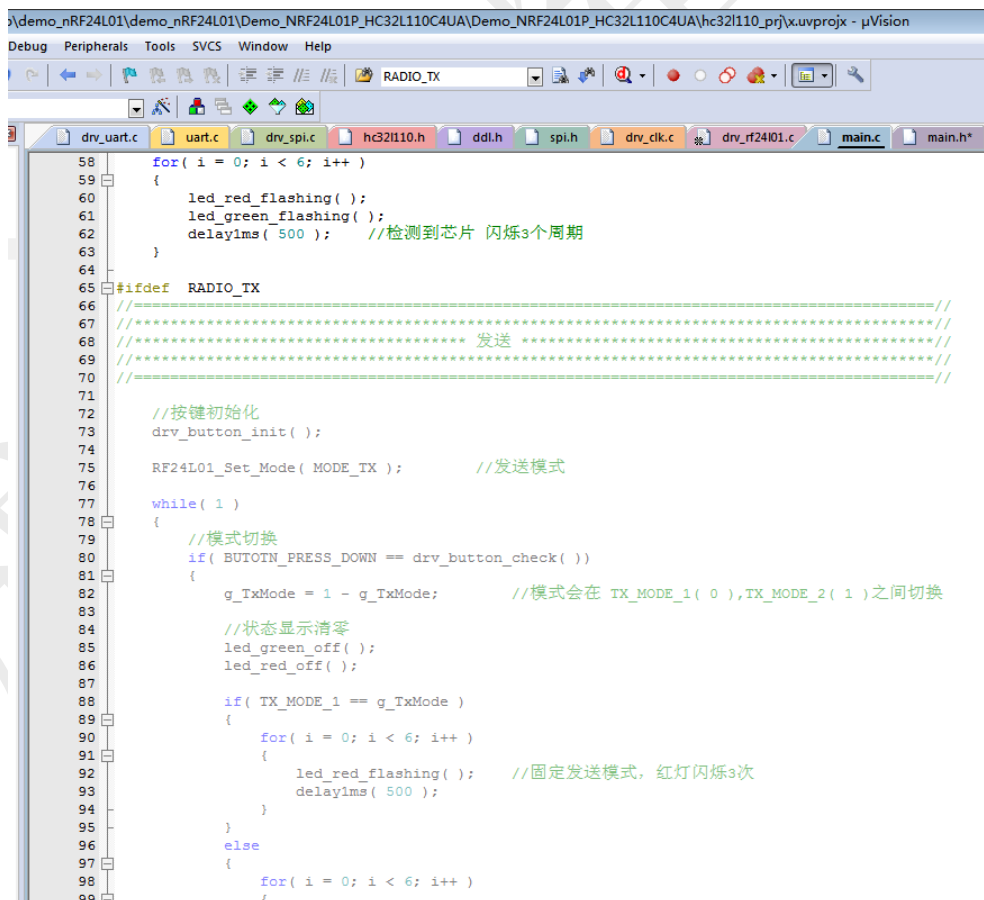
nRF24L01_HC32L110C4UA 的 demo 程序说明

本 demo 程序是基于 HC32L110C4UA 单片机和 NRF24L01 开发设计的。本程序包含了主函数文件 main.c，单片机启动文件 startup_hc32l110.s，以及 SPI 文件 spi.c，基础定时器文件 bt.c，GPIO 文件 gpio.c，串口文件 uart.c 等若干驱动文件。

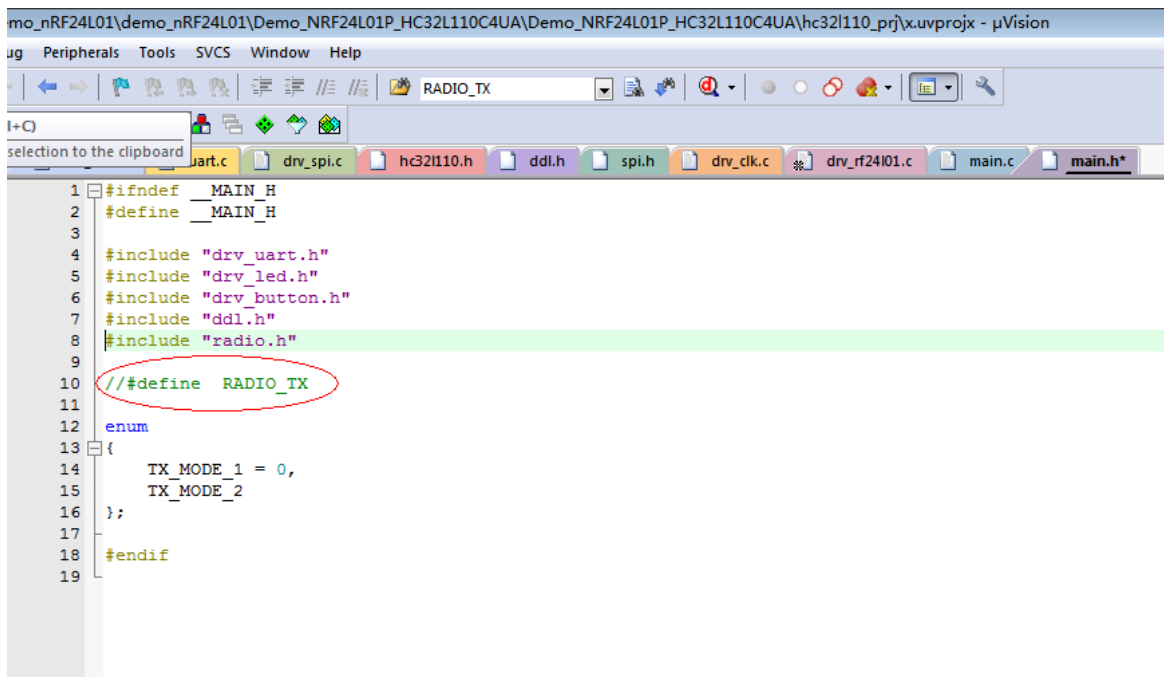
本程序实现的功能是使用 nRF24L01 进行透明传输的功能，但只支持单独的接收或单独的发送。若想实现收发一体的功能需要用户自行修改程序。发送功能中分为了 2 种模式，固定发送模式和自由发送模式，由按键控制。自由发送模式是发送串口收到的数据。

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 文件中可以发现发送功能部分是灰色的，说明该部分是不可编译的，就当前状态进行编译下载，则该模块具有了接收功能。要想编译下载发送功能的程序，需要点开 main.h 文件，将 #define RADIO_TX 释放出来即可。如图：



1) 打开 main.h 文件



```
1 #ifndef MAIN_H
2 #define MAIN_H
3
4 #include "drv_uart.h"
5 #include "drv_led.h"
6 #include "drv_button.h"
7 #include "ddl.h"
8 #include "radio.h"
9
10 // #define RADIO_TX
11
12 enum
13 {
14     TX_MODE_1 = 0,
15     TX_MODE_2
16 };
17
18 #endif
19
```

2) 释放掉圈出部分



```
1 #ifndef MAIN_H
2 #define MAIN_H
3
4 #include "drv_uart.h"
5 #include "drv_led.h"
6 #include "drv_button.h"
7 #include "ddl.h"
8 #include "radio.h"
9
10 #define RADIO_TX
11
12 enum
13 {
14     TX_MODE_1 = 0,
15     TX_MODE_2
16 };
17
18 #endif
19
```

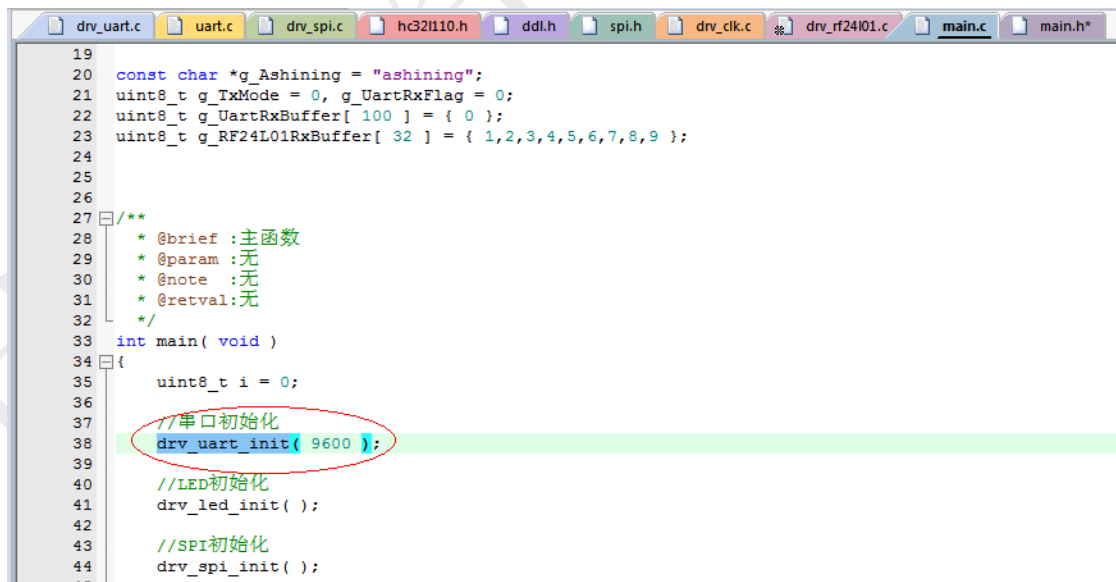
3) 现在的主程序，发送功能可编译，接受功能部分不可编译



```
64
65 #ifndef RADIO_TX
66 //=====
67 //***** 发送 *****
68 //*****
69 //=====
70
71
72 //按键初始化
73 drv_button_init( );
74
75 RF24L01_Set_Mode( MODE_TX ); //发送模式
76
77 while( 1 )
78 {
79     //模式切换
80     if( BUTOTN_PRESS_DOWN == drv_button_check( ) )
81     {
82         g_TxMode = 1 - g_TxMode; //模式会在 TX_MODE_1( 0 ),TX_MODE_2( 1 )之间切换
83
84         //状态显示清零
85         led_green_off( );
86         led_red_off( );
87
88         if( TX_MODE_1 == g_TxMode )
89         {
90             for( i = 0; i < 6; i++ )
91             {
92                 led_red_flashing( ); //固定发送模式，红灯闪烁3次
93                 delaylms( 500 );
94             }
95         }
96     }
97 }
```

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()中的值来更改串口波特率。如图：



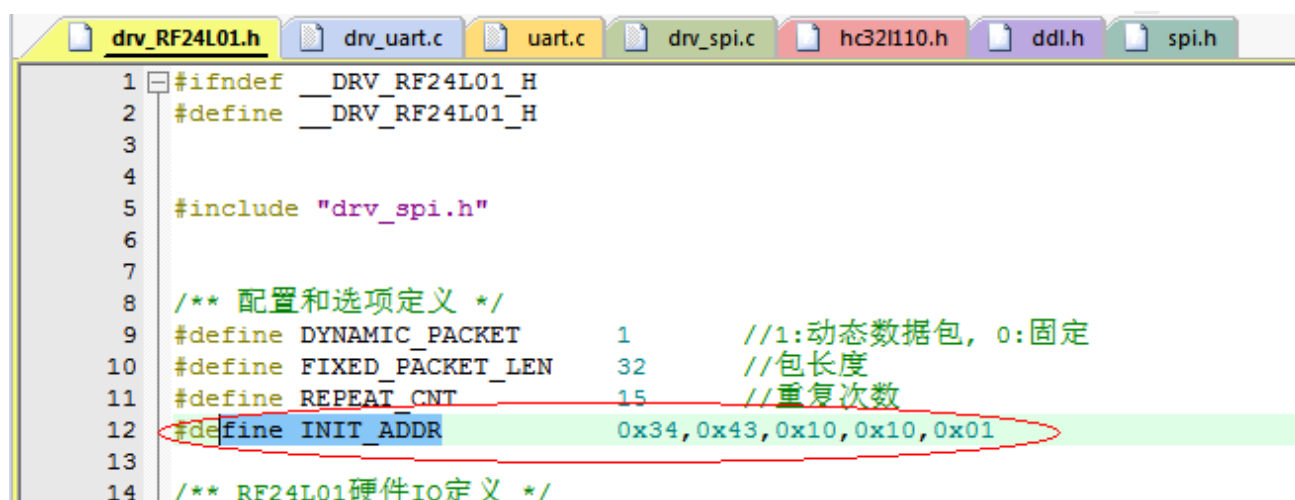
```
19
20 const char *g_Ashining = "ashining";
21 uint8_t g_TxMode = 0, g_UartRxFlag = 0;
22 uint8_t g_UartRxBuffer[ 100 ] = { 0 };
23 uint8_t g_RF24L01RxBuffer[ 32 ] = { 1,2,3,4,5,6,7,8,9 };
24
25
26
27 /**
28  * @brief :主函数
29  * @param :无
30  * @note :无
31  * @retval:无
32  */
33 int main( void )
34 {
35     uint8_t i = 0;
36
37     //串口初始化
38     drv_uart_init( 9600 );
39
40     //LED初始化
41     drv_led_init( );
42
43     //SPI初始化
44     drv_spi_init( );
45 }
```

将图中的圈出来的函数中的 9600 更改为用户自己所需的波特率。如：1200，2400，4800，9600，19200，38400，

57600，115200。

1) 更改 nRF24L01 的通信地址

若需更改 nRF24L01 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。



2) 更改 nRF24L01 的通信配置

若需更改 nRF24L01 的通信配置则需要在 drv_rf24l.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。

```
601  */
602 void RF24L01_Init( void )
603 {
604     uint8_t addr[5] = {INIT_ADDR};
605
606     RF24L01_SET_CE_HIGH( );
607     NRF24L01_Clear_IRQ_Flag( IRQ_ALL );
608     #if DYNAMIC_PACKET == 1
609
610     NRF24L01_Write_Reg( DYNPD, ( 1 << 0 ) ); //使能通道1动态数据长度
611     NRF24L01_Write_Reg( FEATRUE, 0x07 ); //设置特征寄存器, 使能动态负载
612     NRF24L01_Read_Reg( DYNPD ); //读取使能动态负载长度
613     NRF24L01_Read_Reg( FEATRUE ); //读取特征寄存器
614
615     #elif DYNAMIC_PACKET == 0
616
617     L01_WriteSingleReg( L01REG_RX_PW_P0, FIXED_PACKET_LEN ); //固定数据长度
618
619     #endif //DYNAMIC_PACKET
620
621     NRF24L01_Write_Reg( CONFIG, /*( 1 << MASK_RX_DR ) | */ //接收中断
622                         ( 1 << EN_CRC ) | //使能CRC 1个字节
623                         ( 1 << PWR_UP ) ); //开启设备
624     NRF24L01_Write_Reg( EN_AA, ( 1 << ENAA_P0 ) ); //通道0自动应答
625     NRF24L01_Write_Reg( EN_RXADDR, ( 1 << ERX_P0 ) ); //通道0接收
626     NRF24L01_Write_Reg( SETUP_AW, AW_5BYTES ); //地址宽度 5个字节
627     NRF24L01_Write_Reg( SETUP_RETR, ARD_4000US |
628                         ( REPEAT_CNT & 0x0F ) ); //重复等待时间 25
629     NRF24L01_Write_Reg( RF_CH, 60 ); //初始化通道
630     NRF24L01_Write_Reg( RF_SETUP, 0x26 ); //设置通信速度为1
631
632     NRF24L01_Set_TxAddr( &addr[0], 5 ); //设置Tx地址
633     NRF24L01_Set_RxAddr( 0, &addr[0], 5 ); //设置Rx地址
634 }
635
```

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。

```
614
615 if DYNAMIC_PACKET == 0
616
617     L01_WriteSingleReg( L01REG_RX_PW_P0, FIXED_PACKET_LEN );    //固定数据长度
618
619 dif //DYNAMIC_PACKET
620
621 NRF24L01_Write_Reg( CONFIG, /*( 1<<MASK_RX_DR ) |*/          //接收中断
622                      ( 1 << EN_CRC ) |                          //使能CRC 1个字节
623                      ( 1 << PWR_UP ) );                          //开启设备
624 NRF24L01_Write_Reg( EN_AA, ( 1 << ENAA_P0 ) );                //通道0自动应答
625 NRF24L01_Write_Reg( EN_RXADDR, ( 1 << ERX_P0 ) );              //通道0接收
626 NRF24L01_Write_Reg( SETUP_AW, AW_5BYTES );                    //地址宽度 5个字节
627 NRF24L01_Write_Reg( SETUP_RETR, ARD_4000US |                  //重复等待时间 250us
628                      ( REPEAT_CNT & 0x0F ) );
629 NRF24L01_Write_Reg( RF_CH, 60 );                                //初始化通道
630 NRF24L01_Write_Reg( RF_SETUP, 0x26 );                          //设置通信速度为1M
631
632 NRF24L01_Set_TxAddr( &addr[0], 5 );                             //设置TX地址
633 NRF24L01_Set_RxAddr( 0, &addr[0], 5 );                         //设置RX地址
634
635
```

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。