RF Transmitter & Receiver Driver

MIAT-STM32-EVB Development Kit

WU-YANG

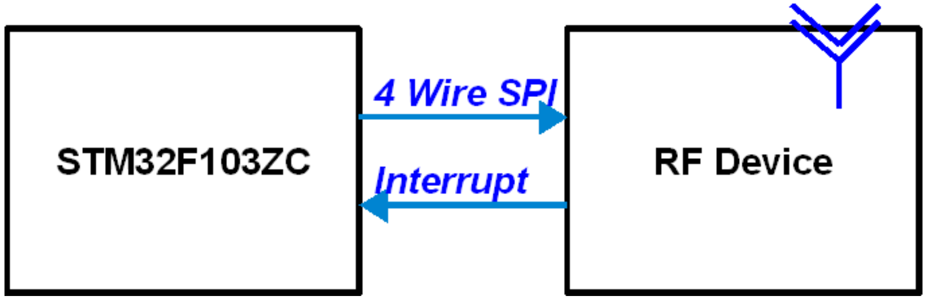
Technology Co., Ltd.

Declared Version

Training Only	
Declare	
Document Version	*1.00*
Release Date	*2009.06.20*
Document Title	*RF Transmitter & Receiver Driver*
Exercise Time	■ *Lecture 25 minutes* ■ *Operating 25 minutes*
Platform	■ *MIAT_STM32* ■ *MIAT_IOB*
Peripheral	■ *RF Device* ■ *RS232 Wire* ■ *USB Wire*
Author	■ *WU-YANG Technology Co., Ltd.*

System Features

The diagram of the system architecture



RF Features

- ■Worldwide 2.4GHz ISM band operation
- ■2Mbps on air data rates

STM32 Features

■SPI Controller

1. 18 Mbits/s
2. The frame is configurable to 8 bits or 16 bits
3. The hardware CRC generation/verification
4. Served by the DMA controller

■GPIO

1. Each of the GPIO pins can be configured by software as output (push-pull or open-drain), as input (with or without pull-up or pull-down) or as peripheral alternate function
2. I/Os on APB2 with up to 18 MHz toggling speed

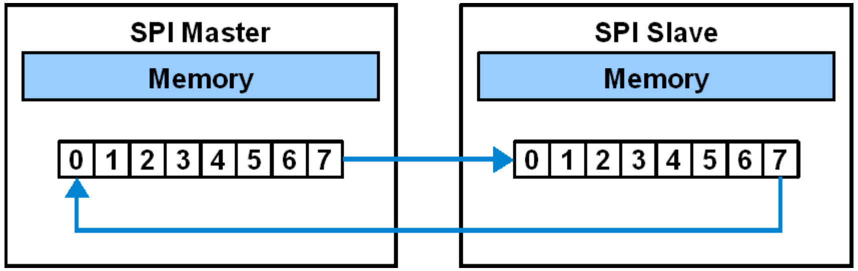
STM32 SPI Bus Controller

Serial Peripheral Interface Bus (SPI)

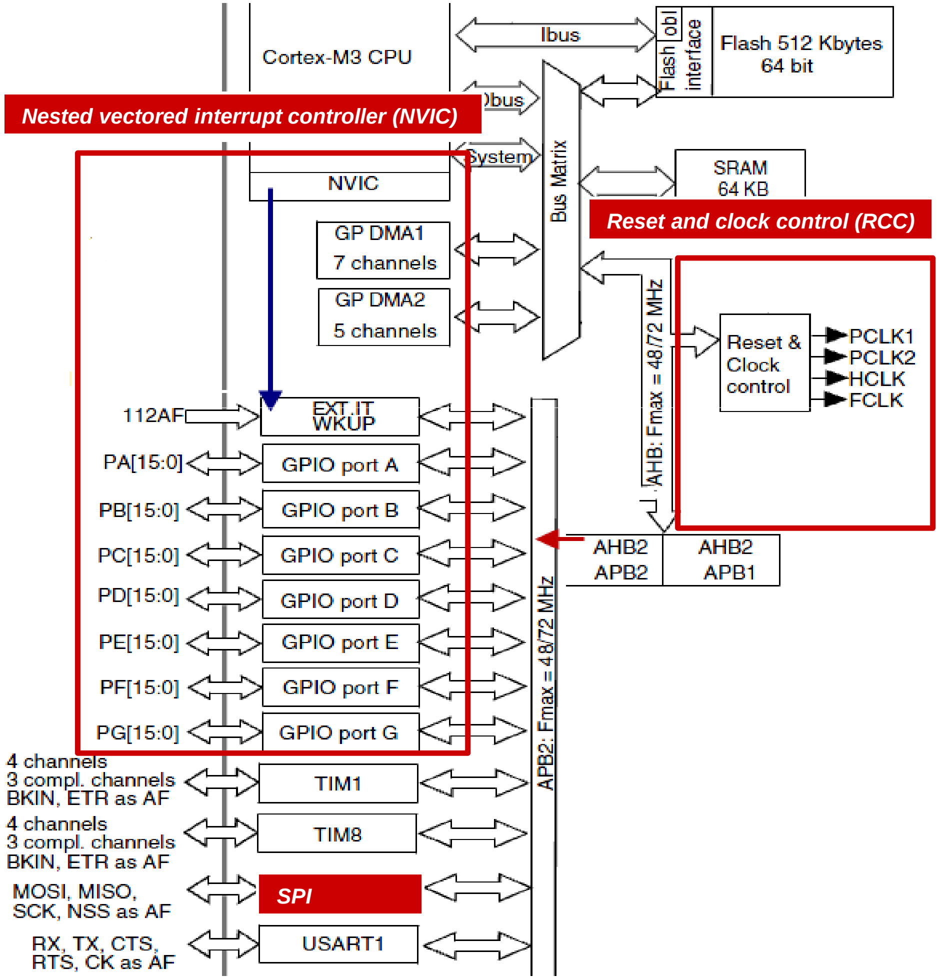
- ***The SPI bus can operate with a single master device and with one or more slave devices***
- ***The SPI bus specifies four logic signals***

a. SCLK — Serial Clock (output from master)
b. MOSI/SIMO — Master Output, Slave Input (output from master)
c. MISO/SOMI — Master Input, Slave Output (output from slave)
d. SS — Slave Select (active low; output from master)

- ***This work operates SPI controller and nRF24L01 on 8 Mhz***

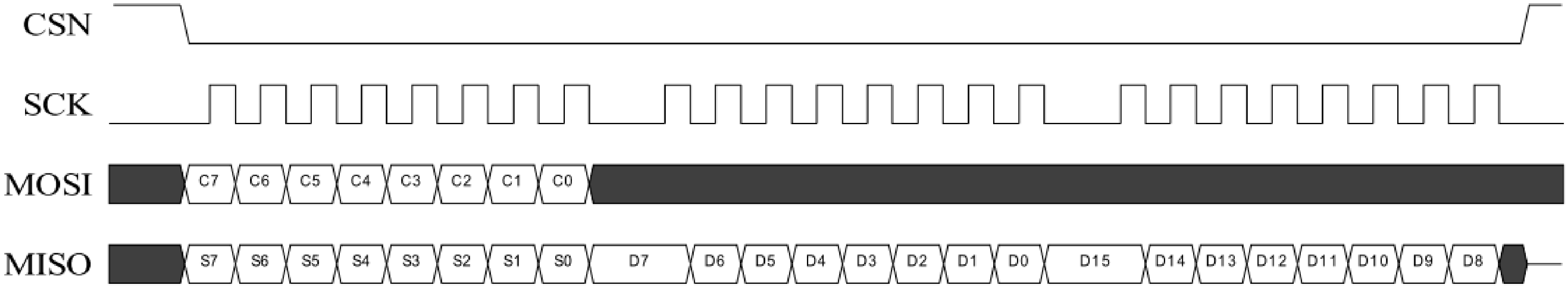


STM32 Architecture Diagram

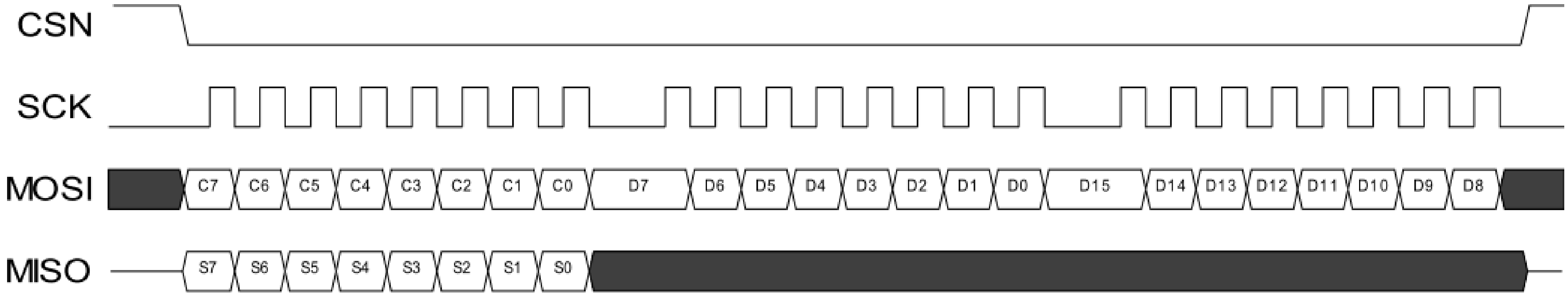


SPI Protocol

SPI – Read Timing Diagram

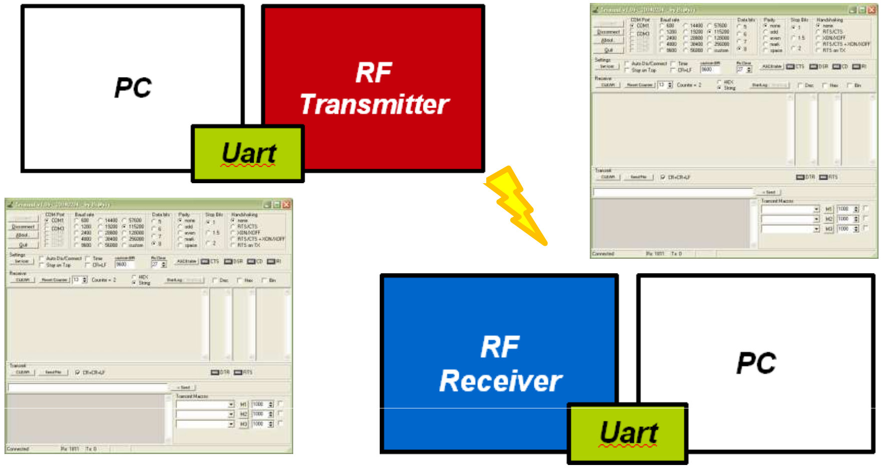

CSN
SCK
MOSI
C7 C6 C5 C4 C3 C2 C1 C0
MISO
S7 S6 S5 S4 S3 S2 S1 S0 D7 D6 D5 D4 D3 D2 D1 D0 D15 D14 D13 D12 D11 D10 D9 D8

SPI – Write Timing Diagram

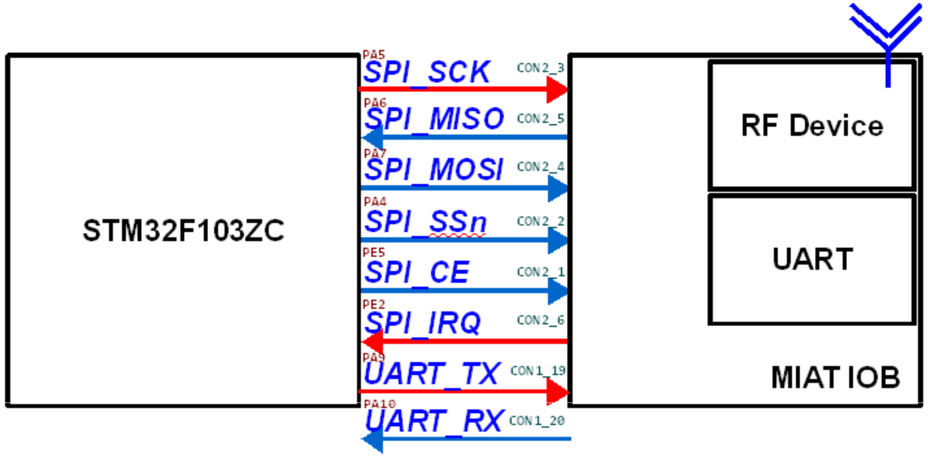

CSN
SCK
MOSI
C7 C6 C5 C4 C3 C2 C1 C0 D7 D6 D5 D4 D3 D2 D1 D0 D15 D14 D13 D12 D11 D10 D9 D8
MISO
S7 S6 S5 S4 S3 S2 S1 S0

Experiment Objective

System Architecture



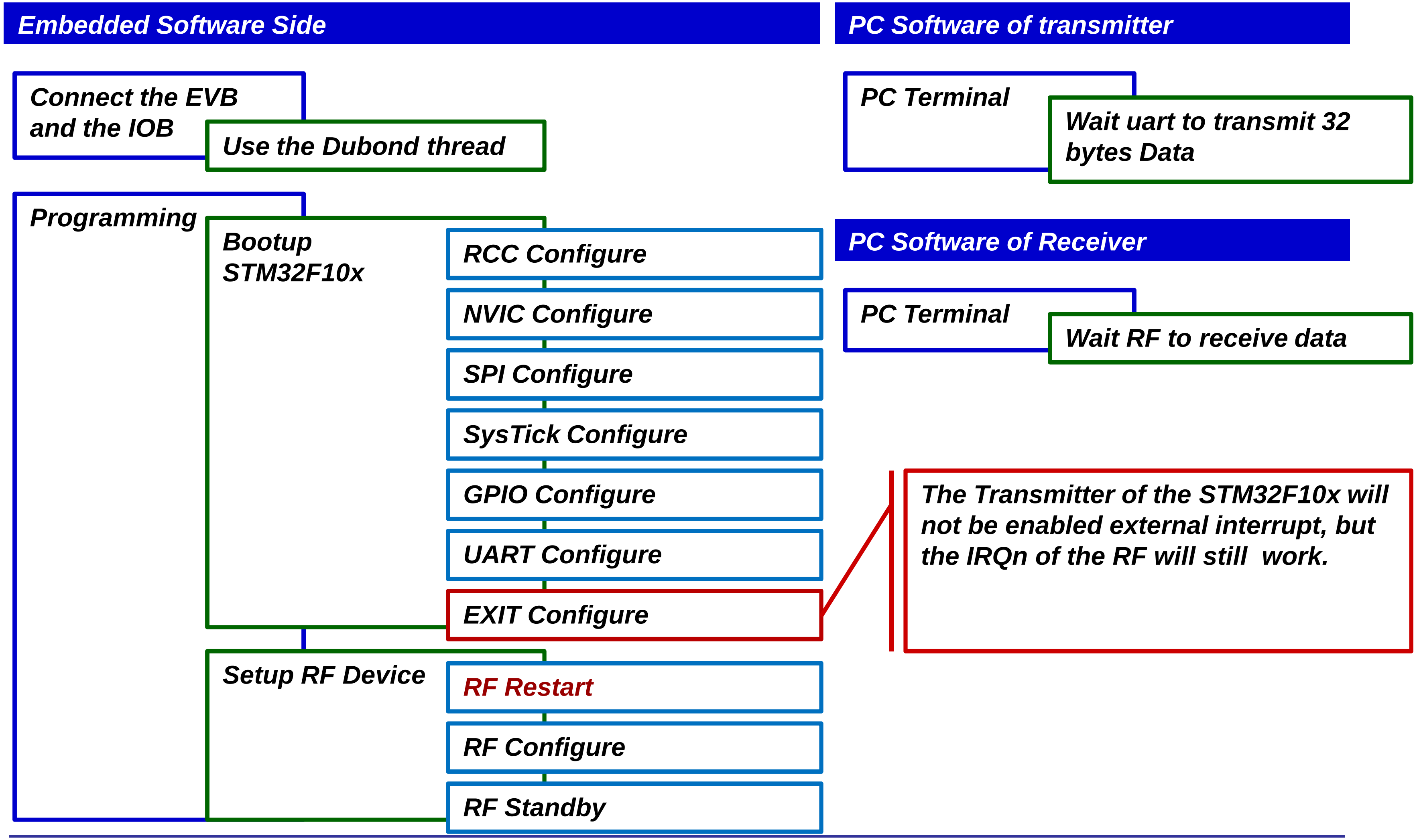
Transmitter & Receiver Architecture Diagram



System Resource

Source	Bus	Function	Signal Name	EVB Location	IOB Location
SPI 1	APB2	SPI	SCK	PA5	CON2_3
	APB2	SPI	MISO	PA6	CON2_5
	APB2	SPI	MOSI	PA7	CON2_4
GPIO A	APB2	GPIO	SSn	PA4	CON2_2
GPIO E	APB2	GPIO	CE	PE5	CON2_1
EXIT 2	APB2	GPIO	IRQn	PE2	CON2_6
UART1	APB2	UART	TX	PA9	CON1_19
	APB2	UART	RX	PA10	CON1_20

Development Flow



Embedded Software Side

Connect the EVB and the IOB

Use the Dubond thread

Programming

Bootup STM32F10x

RCC Configure

NVIC Configure

SPI Configure

SysTick Configure

GPIO Configure

UART Configure

EXIT Configure

Setup RF Device

RF Restart

RF Configure

RF Standby

PC Software of transmitter

PC Terminal

Wait uart to transmit 32 bytes Data

PC Software of Receiver

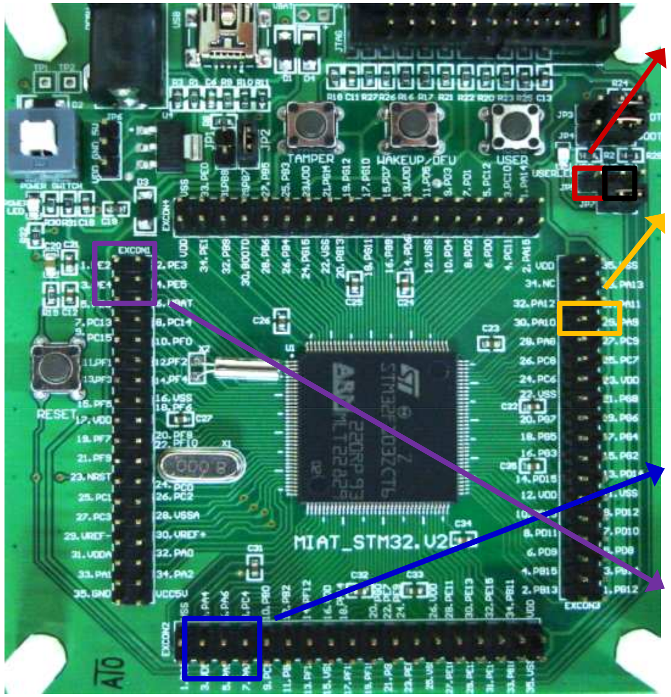
PC Terminal

Wait RF to receive data

The Transmitter of the STM32F10x will not be enabled external interrupt, but the IRQn of the RF will still work.

Floorplanning

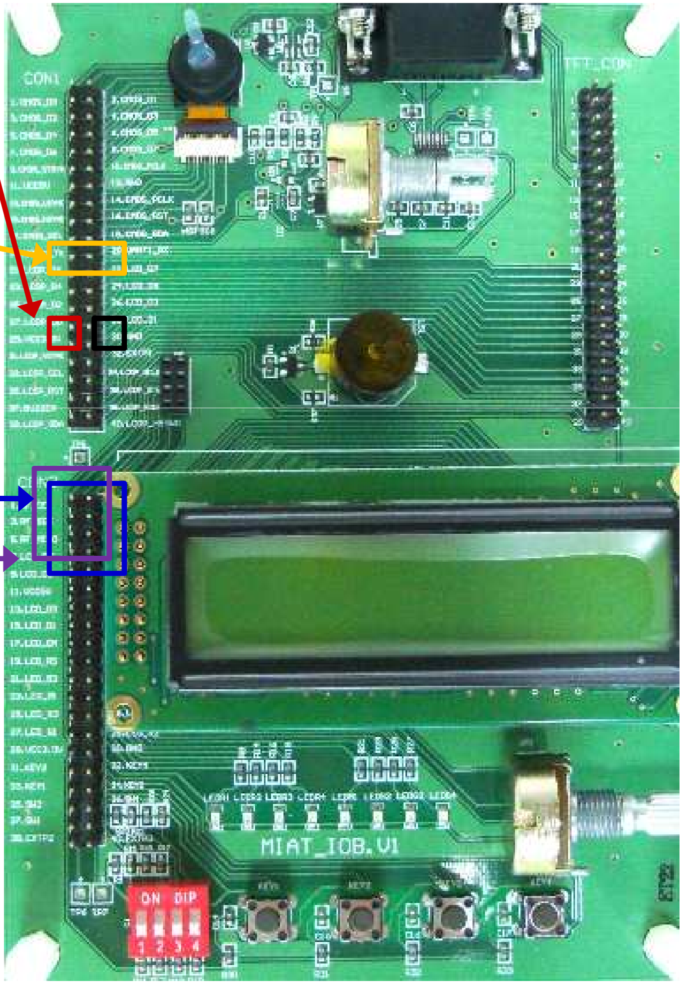
MIAT STM32 Main Board



Pin Mapping

🟥	3V3	JP5	CON1_29
⬛	GND	JP5	CON1_30
🟨	UART TX	PA9	CON1_19
	UART RX	PA10	CON1_20
🟦	SPI CSn	PA4	CON2_2
	SPI SCK	PA5	CON2_3
	SPI MISO	PA6	CON2_5
	SPI MOSI	PA7	CON2_4
🟪	SPI IRQn	PE2	CON2_6
	SPI CE	PE5	CON2_1

MIAT IO Board



RF Device Driver

RFLib Functions List

Function Name	Description
RF_ConfigureMode	Change to configure mode
RF_NormalMode	Change to normal mode
RF_FlushTxBuf	Clear Tx buffer
RF_FlushRxBuf	Clear Rx buffer
RF_TransmitData	Transmit data
RF_ReceiveData	Reveive data
RF_Init	Initial RF device register
RF_IrqStatus	Get RF device irq status

Initial Structure	Description
AddressWidth	The address length of the package
RxAddress	The receiver address
TxAddress	The transmitter address
DataLengthPipe0	The receive data length
Config	Set CRC, power on/off
AutoAck	Enable auto ack
RxPipe	Enable pipe
Retry	Enable retrans
Channel	Set channel
Setup	Set transmitter power
Status	RF status
ObserveTx	Make an overall assessment of the channel quality

```
#include "rf_ctrl.h"

RF_ConfTypeDef rfInitConfigure;
RF_ConfigureMode();
RF_FlushTxBuf();

rfInitConfigure.AddressWidth = 3;
rfInitConfigure.RxAddress[0] = 0xE7;
rfInitConfigure.RxAddress[1] = 0xE7;
rfInitConfigure.RxAddress[2] = 0xE7;
rfInitConfigure.RxAddress[3] = 0xE7;
rfInitConfigure.RxAddress[4] = 0xE7;
rfInitConfigure.TxAddress[0] = 0xE7;
rfInitConfigure.TxAddress[1] = 0xE7;
rfInitConfigure.TxAddress[2] = 0xE7;
rfInitConfigure.TxAddress[3] = 0xE7;
rfInitConfigure.TxAddress[4] = 0xE7;
rfInitConfigure.DataLengthPipe0 = 32;
rfInitConfigure.Config = 0x0e; // tx = 0x0e // rx = 0x0f
rfInitConfigure.AutoAck = 0x01;
rfInitConfigure.RxPipe = 0x01;
rfInitConfigure.Retry = 0x0a;
rfInitConfigure.Channel = 0x09;
rfInitConfigure.Setup = 0x0f;
rfInitConfigure.Status = 0x70;
rfInitConfigure.ObserveTx = 0x00;
RF_Init(&rfInitConfigure);

//RF_TransmitData(rftx_payload);
```

RF Receiver IRQHandler

RFLib Functions List

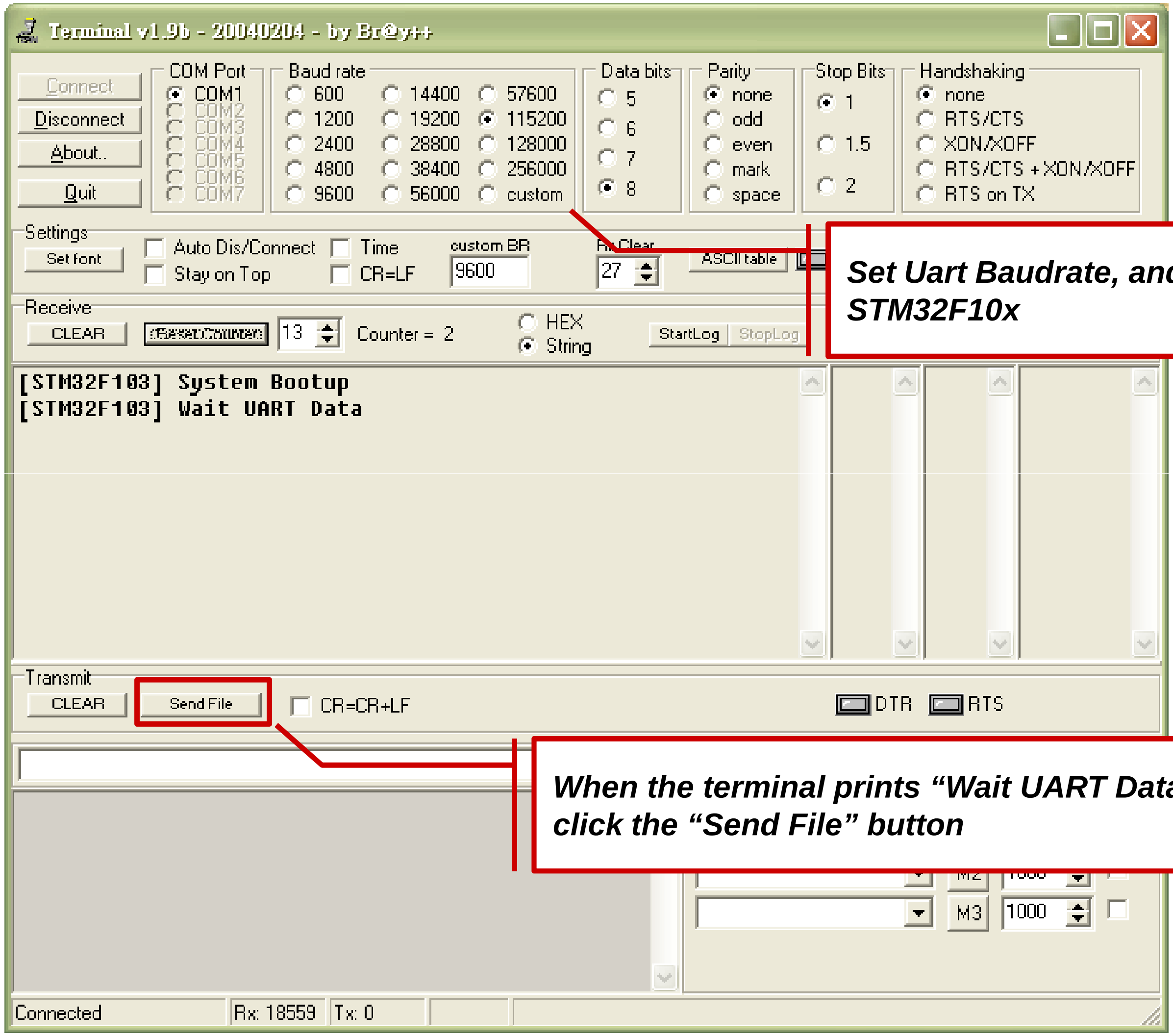
Function Name	Description
RF_ConfigureMode	Change to configure mode
RF_NormalMode	Change to normal mode
RF_FlushTxBuf	Clear Tx buffer
RF_FlushRxBuf	Clear Rx buffer
RF_TransmitData	Transmit data
RF_ReceiveData	Reveive data
RF_Init	Initial RF device register
RF_IrqStatus	Get RF device irq status

```
#include "rf_ctrl.h"

void EXTI2_IRQHandler(void)
{
    int i;
    extern u8 rfrx_payload[32];

    if(EXTI_GetITStatus(EXTI_Line2) != RESET){
        switch(RF_IrqStatus()){
            case 0x00:
                break;
            case 0x10:
                break;
            case 0x20:
                break;
            case 0x40:
                RF_ReceiveData(rfrx_payload);
                    printf("[STM32F103] RF Received: ");
                    for(i=0;i<32;i++) printf("%x ", rfrx_payload[i]);
                    printf("\n\r");
                    break;
        }
        EXTI_ClearITPendingBit(EXTI_Line2);
    }
}
```

RF Transmitter – Uart Terminal



Terminal v1.9b - 20040204 - by Br@y++

COM Port: ● COM1 ○ COM2 ○ COM3 ○ COM4 ○ COM5 ○ COM6 ○ COM7

Baud rate: ○ 600 ○ 1200 ○ 2400 ○ 4800 ○ 9600 ○ 14400 ○ 19200 ○ 28800 ○ 38400 ○ 56000 ○ 57600 ● 115200 ○ 128000 ○ 256000 ○ custom

Data bits: ○ 5 ○ 6 ○ 7 ● 8

Parity: ● none ○ odd ○ even ○ mark ○ space

Stop Bits: ● 1 ○ 1.5 ○ 2

Handshaking: ● none ○ RTS/CTS ○ XON/XOFF ○ RTS/CTS+XON/XOFF ○ RTS on TX

Settings: □ Auto Dis/Connect □ Stay on Top □ Time □ CR=LF custom BR 9600 Rx Clear 27

Receive: 13 Counter = 2 ○ HEX ● String

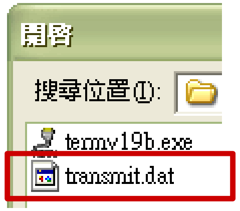
```
[STM32F103] System Bootup
[STM32F103] Wait UART Data
```

Transmit: □ CR=CR+LF

M3 1000

Connected Rx: 18559 Tx: 0

Set Uart Baudrate, and connect to the uart of the STM32F10x



開啟

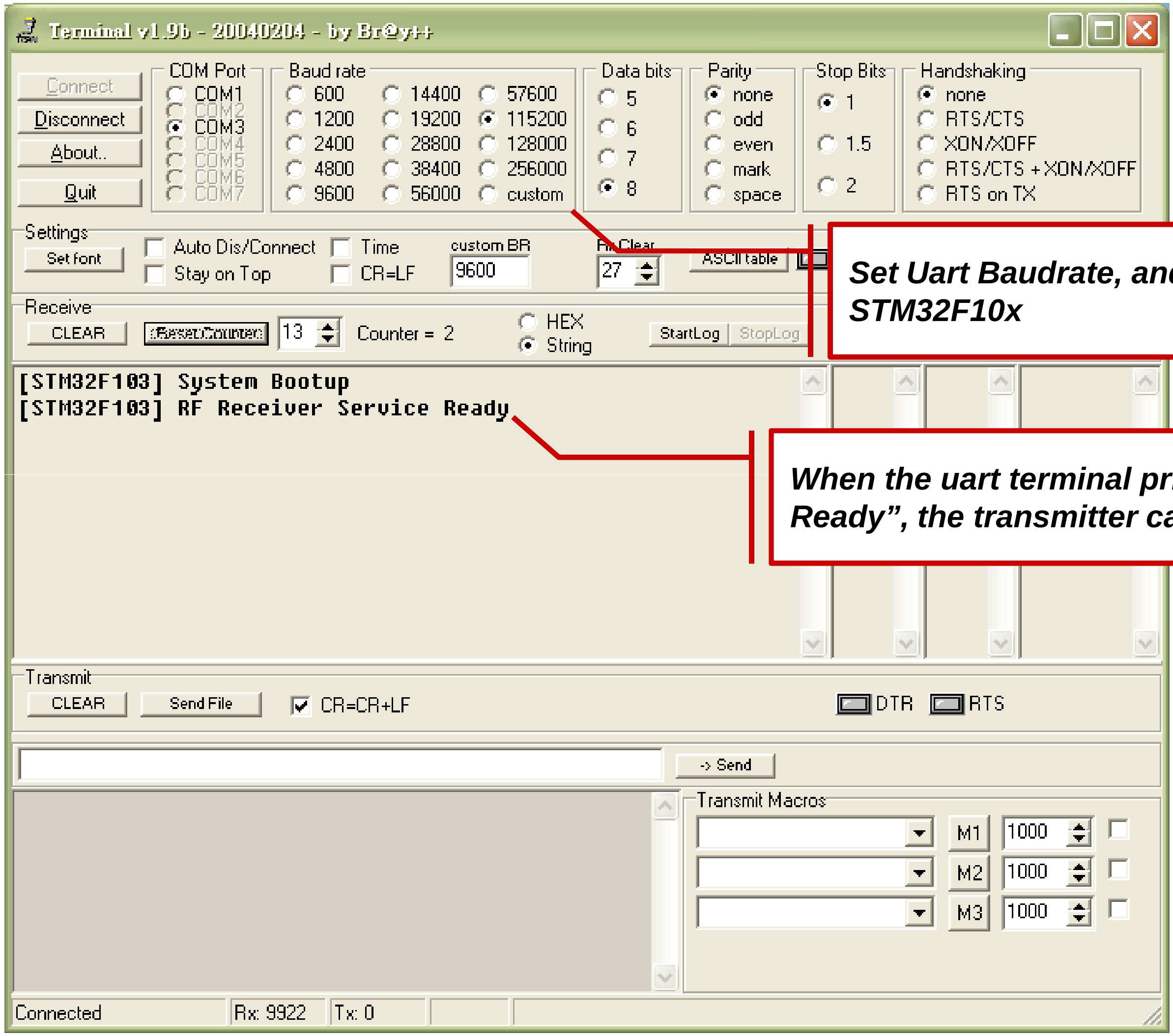
搜尋位置(I):

termv19b.exe

transmit.dat

When the terminal prints “Wait UART Data”, we can click the “Send File” button

RF Receiver – Uart Terminal



Terminal v1.9b - 20040204 - by Br@y++

COM Port: ○ COM1 ● COM3

Baud rate: ○ 600 ○ 1200 ○ 2400 ○ 4800 ○ 9600 ○ 14400 ○ 19200 ○ 28800 ○ 38400 ○ 56000 ○ 57600 ● 115200 ○ 128000 ○ 256000 ○ custom

Data bits: ○ 5 ○ 6 ○ 7 ● 8

Parity: ● none ○ odd ○ even ○ mark ○ space

Stop Bits: ● 1 ○ 1.5 ○ 2

Handshaking: ● none ○ RTS/CTS ○ XON/XOFF ○ RTS/CTS+XON/XOFF ○ RTS on TX

Settings: □ Auto Dis/Connect □ Stay on Top □ Time □ CR=LF custom BR 9600 Rx Clear 27

Receive: 13 Counter = 2 ○ HEX ● String

```
[STM32F103] System Bootup
[STM32F103] RF Receiver Service Ready
```

Transmit: ☑ CR=CR+LF

Transmit Macros: M1 1000 □ M2 1000 □ M3 1000 □

Connected Rx: 9922 Tx: 0

Set Uart Baudrate, and connect to the uart of the STM32F10x

When the uart terminal prints "RF Receiver Service Ready", the transmitter can transmit the data.

Example Document Structure

Document Structure

Folder	File	Description
project	Project.Uv2	Keil RVMDK project file
source	main.c	Main function
	stm32f10x_it.c	Interrupt handle
content	stm32f10x_conf.h	Stm32f10x register mapping define
	stm32f10x_it.h	Interrupt handle header file
	init_sys.h	Initial function header file
	timedelay.h	Delay function header file
	rf_ctrl.h	Rf device driver header file
demo	termv19b.exe	Uart terminal program
	transmit.dat	Data file
image	stm32rfrx.dfu	Image file
	stm32rftx.dfu	Image file

Exercise

- ***Try to change the address of the RF device***
  - ***Hint: If two RF devices have the same Tx pipe address, the data package will be lost on the air.***
- ***Try to change the channel of the RF device***
  - ***RF device can regulate radio channel***
- ***Try to send other data to receiver***

Keep in Mind

- □ ***Use the DfuSe tool to program the flash of the STM32F10x***

Q & A