Bluetooth module

一、Purpose of the experiment

Use the YahboomRobot APP to control the movement status of the car through the Bluetooth 5.0 module.

The status that can be controlled through the app is as follows:

Stop	Forward	Backward
Left	**Right**	**SpinLeft**
SpinRight	**SpeedUp**	**SpeedDown**

二、Experimental principle

Bluetooth module

The Bluetooth module is used as the slave, and the mobile phone APP is used as the host for data transmission.

The Bluetooth module sends the data transmitted by the mobile phone APP to STM32F103RCT6 through TXD, and the single-chip microcomputer then parses the received data to obtain the corresponding instruction data to print relevant control information.

Data format received by STM32F103RCT6 (refer to the communication protocol of the intelligent applet):

Data header + data + data footer
$ + data + #

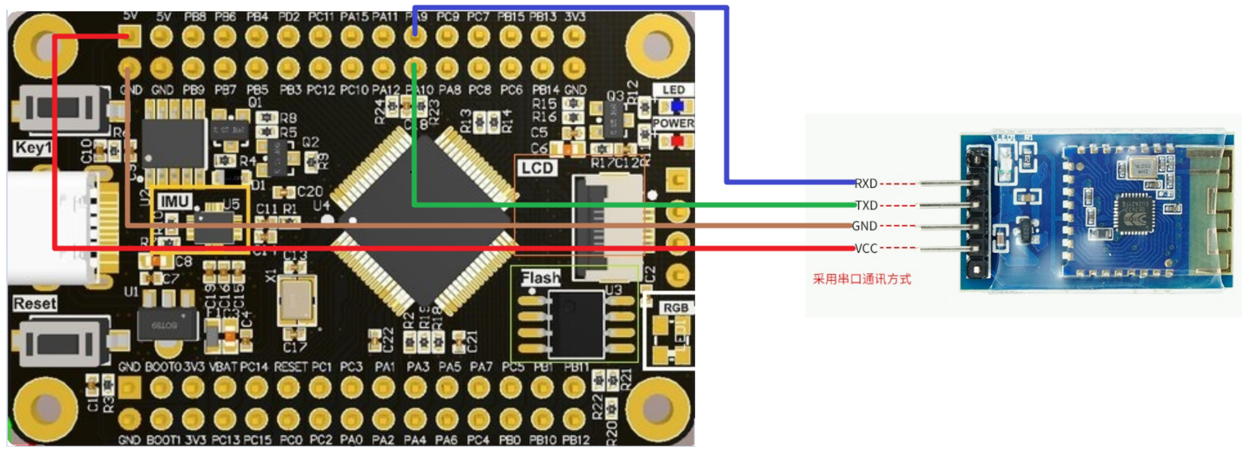
三、Hardware connection

Development board：STM32F103RCT6

Module：Bluetooth 4.0/5.0 module

Connect：DuPont Line (several)

Hardware wiring



Bluetooth module	STM32F103RCT6
TXD	RXD(PA10)
RXD	TXD(PA9)
VCC	5V
GND	GND

Note:

After the wiring is completed, burn the Bluetooth control code (the burning process needs to disconnect the wiring of PA9 and PA10, and then connect the DuPont line after the burning is completed).

四、Main code

1.Implementation principle

Use serial port interrupts to get data and store the correct data in an array.

According to the corresponding position element data in the array to judge the control instructions sent by the APP, you can see the communication protocol, this routine provides a parsing idea, specific optimization and code modification need to be dealt with by themselves.

2.Main code

1.main.c

```
#include "stm32f10x.h"
#include "SysTick.h"
#include "Uart.h"

int main(void)
{
    SysTick_Configuration();
    Uart1_Configuration();
    Uart1_NVIC_Configuration();
    printf("Yahboom!\n");

    while(1)
    {
            APP_Control();
    }
}
```

2.Uart.c

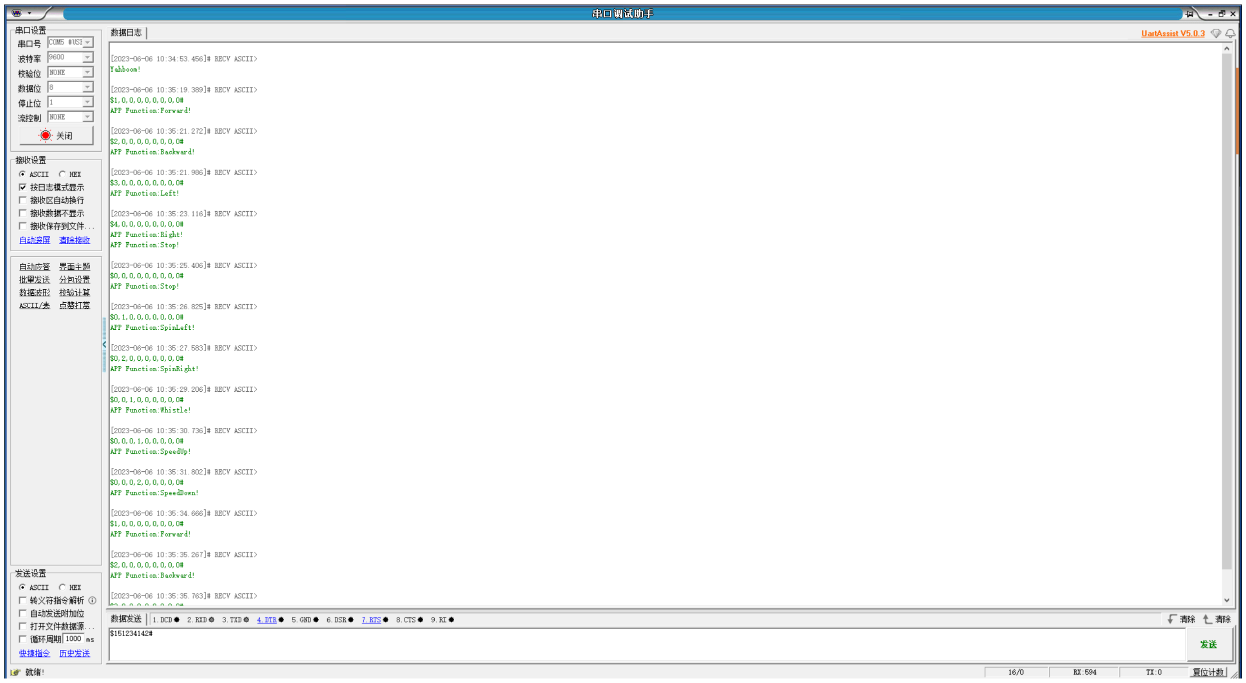
```
#include "Uart.h"

uint8_t rxd_index = 0;
uint8_t rxd_buf[30];
uint8_t rxd_flag = 0;

void Uart1_Configuration(void)
{
    GPIO_InitTypeDef GPIO_InitStructure;
    USART_InitTypeDef USART_InitStructure;

    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA | RCC_APB2Periph_USART1, ENABLE);

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_9; //TX
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10; //RX
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

    USART_InitStructure.USART_BaudRate = 9600;
    USART_InitStructure.USART_WordLength = USART_WordLength_8b;
    USART_InitStructure.USART_StopBits = USART_StopBits_1;
    USART_InitStructure.USART_Parity = USART_Parity_No;
    USART_InitStructure.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
    USART_InitStructure.USART_Mode = USART_Mode_Tx | USART_Mode_Rx;
    USART_Init(USART1, &USART_InitStructure);

    USART_ITConfig(USART1, USART_IT_RXNE, ENABLE); //接收中断

    USART_Cmd(USART1, ENABLE);
}

void Uart1_NVIC_Configuration(void)
{
    NVIC_InitTypeDef NVIC_InitStructure;

    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_1);
    NVIC_InitStructure.NVIC_IRQChannel = USART1_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 1;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);

}

int fputc(int c, FILE *pt)
{
    USART_TypeDef* USARTx = USART1;
    while(USART_GetFlagStatus(USARTx, USART_FLAG_TXE) == RESET);
    USART_SendData(USARTx, c);
    while(USART_GetFlagStatus(USARTx, USART_FLAG_TC) == RESET);

    return 0;
}

int fgetc(FILE *pt)
{
    USART_TypeDef* USARTx = USART1;
    while(USART_GetFlagStatus(USARTx, USART_FLAG_RXNE) == RESET);
    return (int)USART_ReceiveData(USARTx);
}

void Rec_Data(void)
{
        if(rxd_flag == 1)
        {
            printf("$");
            printf("%s",rxd_buf);
            printf("#");
            printf("\n");
            rxd_flag = 0;
        }
}

void APP_Control(void)
{
    if(rxd_flag == 1)
    {
        switch(rxd_buf[0])
        {
            case '1':
                Rec_Data();
            printf("APP Function:Forward!\n");
                break;
            case '2':
                Rec_Data();
                printf("APP Function:Backward!\n");
                break;
            case '3':
                Rec_Data();
                printf("APP Function:Left!\n");
                break;
            case '4':
                Rec_Data();
                printf("APP Function:Right!\n");
                break;
        }
        switch(rxd_buf[2])
        {
            case '1':
                Rec_Data();
                printf("APP Function:SpinLeft!\n");
                break;
            case '2':
                Rec_Data();
                printf("APP Function:SpinRight!\n");
                break;
        }
        switch(rxd_buf[4])
        {
            case '1':
                Rec_Data();
                printf("APP Function:Whistle!\n");
                break;
        }
        switch(rxd_buf[6])
        {
            case '1':
                Rec_Data();
                printf("APP Function:SpeedUp!\n");
                break;
            case '2':
                Rec_Data();
                printf("APP Function:SpeedDown!\n");
                break;
        }
        if((rxd_buf[0] == '0') && (rxd_buf[2] == '0') && (rxd_buf[4] == '0') &&
(rxd_buf[6] == '0') && (rxd_buf[8] == '0') && (rxd_buf[10] == '0') && (rxd_buf[12]
== '0') && (rxd_buf[14] == '0') && (rxd_buf[16] == '0'))
        {
            Rec_Data();
            printf("APP Function:Stop!\n");
        }
        rxd_flag = 0;
    }

}


void USART1_IRQHandler(void)
{
    uint8_t recv_dat = 0;
    static uint8_t rec_state = 0;
    while(USART_GetFlagStatus(USART1, USART_FLAG_RXNE) == SET)
    {
        recv_dat = USART_ReceiveData(USART1);

        switch(rec_state)
        {
        case 0:
            if((recv_dat == '$') && (!rxd_flag))
            {
                rec_state = 1;
                rxd_index = 0;
            }
            else
            {
                rec_state = 0;
            }
            break;
        case 1:
            if(recv_dat == '#')
            {
                rxd_flag = 1;
                rec_state = 0;
            }
            else
            {
                rxd_buf[rxd_index++] = recv_dat;
            }
            break;
        }
    }
}
```

3.Uart.h

```
#ifndef __UART_H__
#define __UART_H__

#include "stm32f10x.h"
#include "stdio.h"

extern uint8_t rxd_flag;
extern uint8_t rxd_index;
extern uint8_t rxd_buf[30];

void Uart1_Configuration(void);
void Uart1_NVIC_Configuration(void);
void APP_Control(void);
void Rec_Data(void);

int fputc (int c, FILE *pt);
int fgetc(FILE *pt);

#endif
```

五、Phenomenon

Use the Bluetooth connection in the APP to control the trolley, and the serial port will continuously print the current status information of the trolley.



串口设置
串口号: COM5 #USI
波特率: 9600
校验位: NONE
数据位: 8
停止位: 1
流控制: NONE

接收设置
[✓ ASCII] HEX
[✓] 按日志模式显示
☐ 接收区自动换行
☐ 接收数据不显示
☐ 接收保存到文件...

发送设置
[✓ ASCII] HEX
☐ 转义符指令解析
☐ 自动发送附加位
☐ 打开文件数据源...
☐ 循环周期 1000 ms

```
[2023-06-06 10:34:53.456]# RECV ASCII>
Yahboom!

[2023-06-06 10:35:19.389]# RECV ASCII>
$1,0,0,0,0,0,0,0,0#
APP Function:Forward!

[2023-06-06 10:35:21.272]# RECV ASCII>
$2,0,0,0,0,0,0,0,0#
APP Function:Backward!

[2023-06-06 10:35:21.986]# RECV ASCII>
$3,0,0,0,0,0,0,0,0#
APP Function:Left!

[2023-06-06 10:35:23.116]# RECV ASCII>
$4,0,0,0,0,0,0,0,0#
APP Function:Right!
APP Function:Stop!

[2023-06-06 10:35:25.406]# RECV ASCII>
$0,0,0,0,0,0,0,0,0#
APP Function:Stop!

[2023-06-06 10:35:26.825]# RECV ASCII>
$0,1,0,0,0,0,0,0,0#
APP Function:SpinLeft!

[2023-06-06 10:35:27.583]# RECV ASCII>
$0,2,0,0,0,0,0,0,0#
APP Function:SpinRight!

[2023-06-06 10:35:29.206]# RECV ASCII>
$0,0,1,0,0,0,0,0,0#
APP Function:Whistle!

[2023-06-06 10:35:30.736]# RECV ASCII>
$0,0,0,1,0,0,0,0,0#
APP Function:SpeedUp!

[2023-06-06 10:35:31.802]# RECV ASCII>
$0,0,0,2,0,0,0,0,0#
APP Function:SpeedDown!

[2023-06-06 10:35:34.666]# RECV ASCII>
$1,0,0,0,0,0,0,0,0#
APP Function:Forward!

[2023-06-06 10:35:35.267]# RECV ASCII>
$2,0,0,0,0,0,0,0,0#
APP Function:Backward!

[2023-06-06 10:35:35.763]# RECV ASCII>
```

```
$151234142#
```

16/0 RX:594 TX:0