目录

通信协议20190722

SPI总线

四线双工
速度不超过30Mbps
CPOL=0 CPHA=1
8位数据，MSB先行
受STM32限制，字节间隔不小于30us
STM32为从机，树莓派为主机

数据格式

- 使用ASCII字符串传输
- 树莓派发送如下字符串为一帧
    - <123,456,3234,78,123>
    - 从第一个字符`<`向传输至`>`结束
    - 两帧之间间隔不小于回调函数用时+字节间隔
- 数据以十进制字符串表达
    - 32位无符号数
    - 前导零随意，如`<00023,234,08,007>`
    - 0可省略，如`<123,456,,888,,111>`
    - 数量可变，不超过255个
    - 英文逗号分隔，无空格
- 树莓派发送一个字符同时接收来自STM32的返回的字节，非ASCII码

异常和纠错

- STM32未能接收字节
    1. STM32输出错误码0到树莓派
    2. 树莓派立即停止传输
    3. 树莓派重新发送整个帧
- 数据错误
    1. STM32输出错误码1或2到树莓派
    2. 树莓派立即停止传输
    3. 树莓派重新发送整个帧
- 减少首字节干扰（不必须）
    1. 树莓派发送首字节`<`

2. STM32输出7到树莓派
3. 树莓派重新发送首字符`<`
4. 树莓派确认STM32输出字节0x34
5. 树莓派继续发送后续内容

- 正常情况

1. 树莓派发送一个字节
2. 树莓派同时收到一个字节0x34

检验

> 下面的符号`-`表示字节0x34
> 符号`x`由上一次最后一个字符确定
> STM32返回的数字不是ASCII码

例1 首次发送

```
发送 <<123,45,4374,7ad,98>
返回 7-------------------
STM32实际接收123,45,4374,7,98
```

例2 省略数据

```
发送 <123,33,025,,324>
返回 x-----------------
STM32实际接收123,33,25,0,324
```

例3 空格、字母和符号

```
发送 <123,3v3,^y5ab ,f$%,324>
返回 x------1--11-111-111----
STM32实际接收123,33,5,0,324
```

例4

```
发送 <1,a><2,5>
返回 x---1-----
STM32实际接收1,0和2,5
```

例5 不完整帧

```
发送 13,42><2,3,4>
返回 x222222------
STM32实际接收2,3,4
若补全了上一次发送最后不完整的帧，则接收到...13,42和2,3,4，见例9
```

例6 帧外的字节

```
发送 <12>a<2,3>
返回 x----2----
STM32实际接收12和2,3
```

例7 乱码

```
发送 1a;'.[&*
返回 x2222222
```

例8 `>` 和 `<` 间隔太短

```
发送 <1,2><3,4,5>
返回 x----0000000
STM32实际接收1,2
```

例9 不完整帧

```
发送 <1,2><3,5,3
返回 x----------
STM32实际接收1,2，若下一次发送补全不完整的帧，将接收到3,5,3...，见例5
```

代码

树莓派Raspbian Scretch

```
1  import spidev
2  import time
3  import sys
4  import threading
5
6
```

```
class Thth(threading.Thread):
    '''
    daemon class
    '''
    input_str = ""
    def __init__(self):
        threading.Thread.__init__(self)

    def run(self):
        while self.input_str != "q":
            self.input_str = str(sys.stdin.readline()).strip("\n")
        print("exiting...")


def fun():
    spi = spidev.SpiDev()
    spi.open(0, 0)  # /dev/spidev0.0
    spi.max_speed_hz = 10000000  # 10MHz
    spi.mode = 0b01  # CPOL=0 CPHA=1
    spi.bits_per_word = 8

    to_send = "<12,34,56,78,6899,1234>"
    i = 0
    thth = Thth()
    thth.start()  # start daemon thread

    while True:
        # press q to exit
        if thth.input_str == "q":
            break
        li = []
        i += 1  # count
        for ch in to_send:
            result = spi.xfer([ord(ch)])
            li.append(result[0])  # store MISO data

        print(" %dth send:\t" % i, to_send, sep="")
        print(" status:    \t", end="")
        for item in li:
            if item == 0x34:  # spi works
                print("-", end="")
            else:  # print error code
                print(item, end="")
        print("\n")
        print("q and enter to exit")
        time.sleep(0.5)

    thth.join()
    spi.close()

if __name__ == "__main__":
   fun()
```

STM32F407裸机

```
#include "stm32f4xx.h"
#include "utils.h"
#include "bsp_spi.h"

void aaa(uint32_t* data, uint8_t length)
{
```

```
  uint8_t i;
  //打印读到的数据
  printf("<");
  for (i = 0; i < length; i++)
  {
    printf("%d", data[i]);
    if (i < length - 1)
      printf(", ");
  }
  printf(">\r\n");
}

/**
 * @brief entry~
 */
int main()
{
  UTILS_InitDelay();
  UTILS_InitUart(115200);
  UTILS_DelayMs(100);
  printf("ahb=%d, apb1=%d, apb2=%d\r\n", AhbClock, Apb1Clock, Apb2Clock);
  BSP_SPI_Init(aaa);
  while (1)
  {

  }
}
```