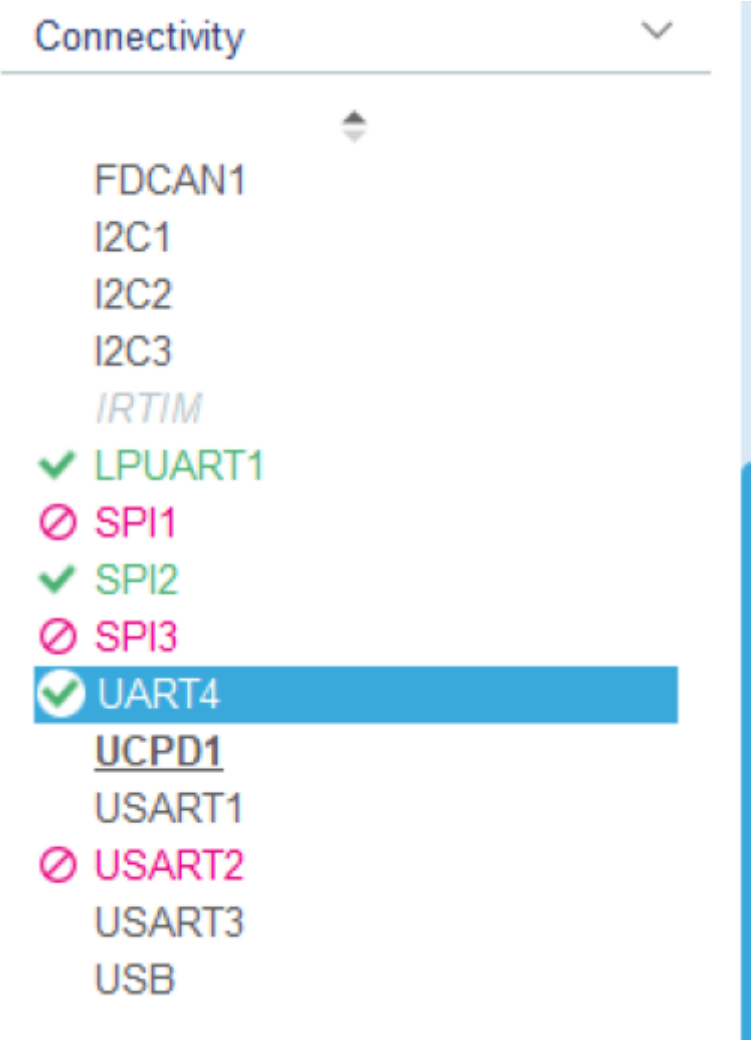


实验4： UART/SPI/IIC通信

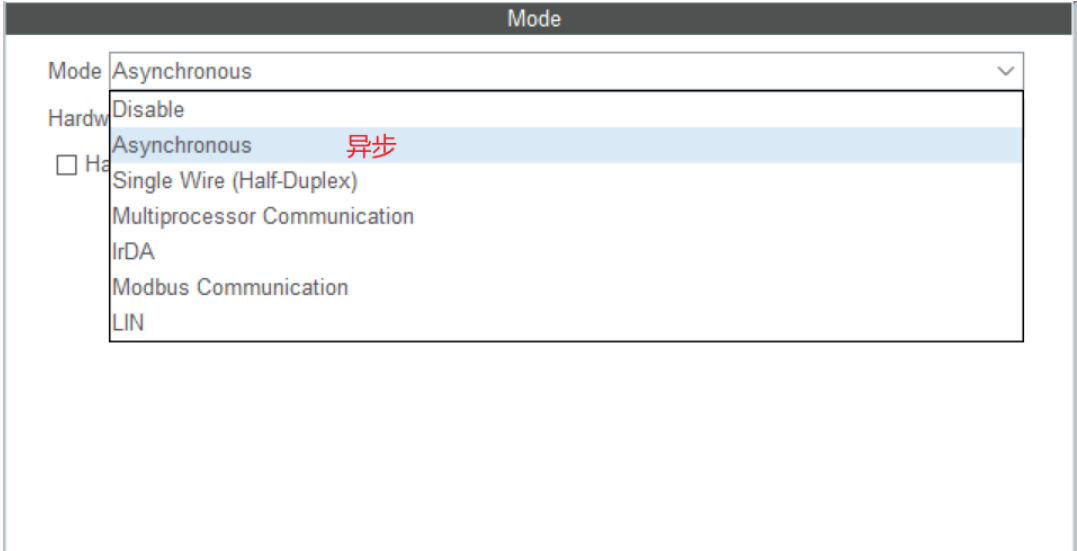
1. UART+DMA

1. Cubemx配置

1. 选择Connectivity其中可选的UART，以UART4为例，



2. 选择异步模式



3. 基本参数配置，例如：115200，8位，无奇偶校验位，一个停止位等


```

6  {
7      int handle;
8  };                                     // 标准库
    需要的支持函数
9
10 FILE __stdout;                         // FILE
    在stdio.h文件
11 void _sys_exit(int x)
12 {
13     x = x;                             // 定义
    _sys_exit()以避免使用半主机模式
14 }
15 int fputc(int ch, FILE *f)             // 重写
    fputc函数, 使printf的输出由UART1实现, 这里使用USART1
16 {
17     // 注意, 不能使用HAL_UART_Transmit_IT(), 机制上会冲突; 因为调用中断发送
    函数后, 如果上次发送还在进行, 就会直接返回! 它不会继续等待, 也不会数据填入队列排队
    发送
18     HAL_UART_Transmit(&huart4, (uint8_t *)&ch, 1, 0x02); // 使用
    HAL_UART_Transmit, 相等于是USART4->DR = ch, 函数内部加了简单的超时判断(ms),
    防止卡死
19     return ch;
20 }

```

8. 定义结构体方便数据收发

```

1  typedef struct{
2      uint16_t ReceiveNum;
3      uint8_t ReceiveData[512];
4      uint8_t BuffTemp[512];
5  }USART_Typedef;

```

9. 重写DMA中断函数

```

1  void HAL_UARTEx_RxEventCallback(UART_HandleTypeDef *huart, uint16_t
    Size)
2  {
3      if (huart == &huart4) // 判断串口
4      {
5          __HAL_UNLOCK(huart); // 解锁串口状态
6
7          USART4.ReceiveNum = Size; // 把接收字节数, 存入结构体
            USART4.ReceiveNum, 以备使用
8          memset(USART4.ReceiveData, 0, sizeof(USART4.ReceiveData)); //
            清0前一帧的接收数据
9          memcpy(USART4.ReceiveData, USART4.BuffTemp, Size); // 存储数据
            到USART4.ReceiveData[]
10         HAL_UARTEx_ReceiveToIdle_DMA(&huart4, USART4.BuffTemp,
            sizeof(USART4.BuffTemp)); // 再次开启DMA空闲中断; 每当接收完指定长度, 或者
            产生空闲中断时, 就会来到这个
11     }
12 }

```

10. 一个随意的主程序 (能完成基本通信展示作用即可)

```

1   while (1)
2   {
3
4       HAL_UARTEx_ReceiveToIdle_DMA(&huart4, USART4.BuffTemp, sizeof(USART4.BuffTemp));
5
6       if (USART4.ReceiveNum > 0)
7       {
8           int num1, num2;
9           sscanf((char*)USART4.ReceiveData, "%d %d", &num1,
10              &num2);
11           num1++; num2++;
12           char sendData[512];
13           snprintf(sendData, sizeof(sendData), "%d %d", num1,
14              num2);
15           printf("%s ", sendData);
16           USART4.ReceiveNum = 0;
17           memset(USART4.ReceiveData, 0,
18              sizeof(USART4.ReceiveData));
19       }
20       HAL_Delay(1000);
21   }

```

注：引脚可在datasheet查看

即提供的 [um2505-stm32g4-nucleo64-boards-mb1367-stmicroelectronics.pdf](#)

可能使用的库函数

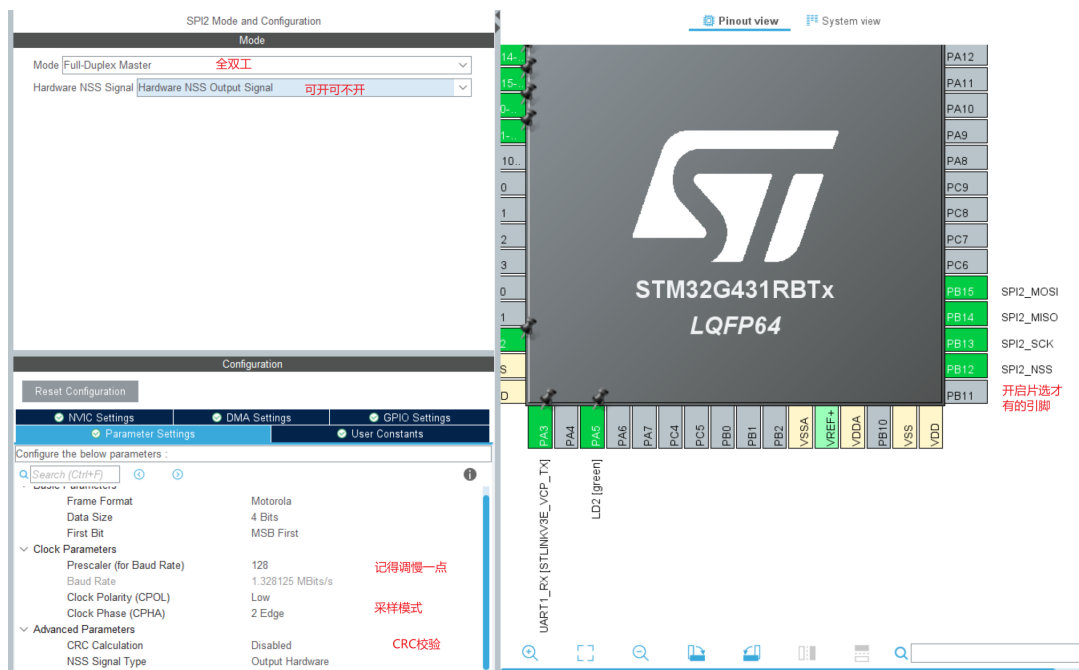
1. `HAL_UART_Transmit()`：这个函数用于发送数据。接口参数主要包括UART_HandleTypeDef结构体指针、要发送的数据缓冲区指针、要发送的数据字节数、超时时间。
2. `HAL_UART_Receive()`：这个函数用于接收数据。接口参数主要包括UART_HandleTypeDef结构体指针、要接收的数据缓冲区指针、要接收的数据字节数、超时时间。
3. `HAL_UART_Transmit_IT()`：这个函数用于中断方式发送数据。参数与 `HAL_UART_Transmit()` 相同。
4. `HAL_UART_Receive_IT()`：这个函数用于中断方式接收数据。参数与 `HAL_UART_Receive()` 相同。
5. `HAL_UART_Transmit_DMA()`：这个函数用于DMA方式发送数据。参数与 `HAL_UART_Transmit()` 相同。
6. `HAL_UART_Receive_DMA()`：这个函数用于DMA方式接收数据。参数与 `HAL_UART_Receive()` 相同。

2. SPI

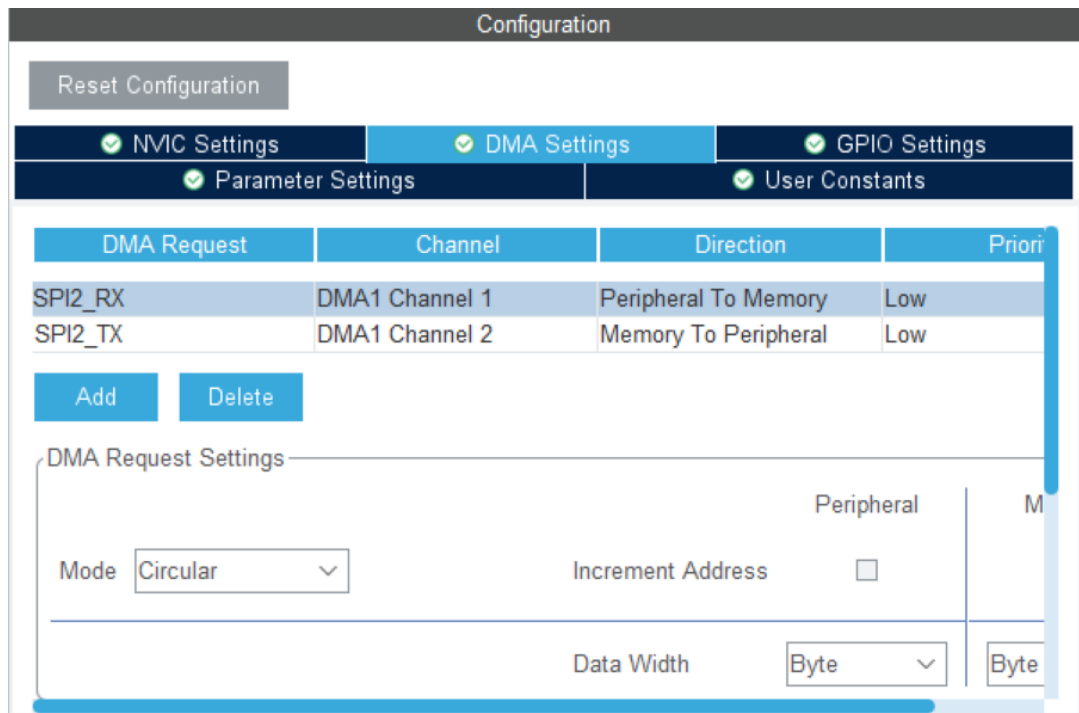
可参考CSDN：[STM32 SPI 双机通信，SPI从机模式使用](#)

1. Cubemx配置

1. 配置SPI



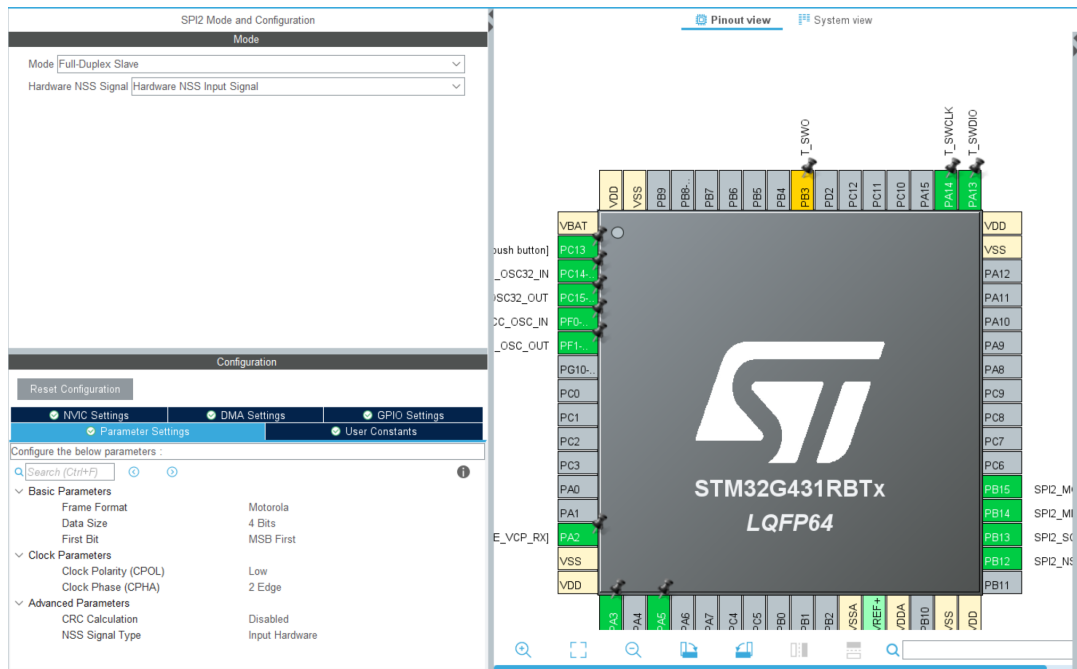
2. 配置DMA



3. 配置中断

Parameter Settings	User Constants		
NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priority
DMA1 channel1 global interrupt	<input checked="" type="checkbox"/>	0	0
DMA1 channel2 global interrupt	<input checked="" type="checkbox"/>	0	0
SPI2 global interrupt	<input checked="" type="checkbox"/>	0	0

4. 从机配置参数记得与主机一致



可能使用的库函数

函数名称	描述	参数
<code>HAL_SPI_Transmit</code>	同步发送数据	<ul style="list-style-type: none"> - <code>SPI_HandleTypeDef</code> - <code>*hspi</code>: SPI句柄 - <code>uint8_t *pData</code>: 发送数据缓冲区 - <code>uint16_t Size</code>: 发送数据大小 - <code>uint32_t Timeout</code>: 超时时间
<code>HAL_SPI_Receive</code>	同步接收数据	<ul style="list-style-type: none"> - <code>SPI_HandleTypeDef</code> - <code>*hspi</code>: SPI句柄 - <code>uint8_t *pData</code>: 接收数据缓冲区 - <code>uint16_t Size</code>: 接收数据大小 - <code>uint32_t Timeout</code>: 超时时间
<code>HAL_SPI_TransmitReceive</code>	同步发送和接收数据	<ul style="list-style-type: none"> - <code>SPI_HandleTypeDef</code> - <code>*hspi</code>: SPI句柄 - <code>uint8_t *pTxData</code>: 发送数据缓冲区 - <code>uint8_t *pRxData</code>: 接收数据缓冲区 - <code>uint16_t Size</code>: 数据大小 - <code>uint32_t Timeout</code>: 超时时间

函数名称	描述	参数
<code>HAL_SPI_Transmit_IT</code>	中断模式发送数据	<ul style="list-style-type: none"> - <code>SPI_HandleTypeDef</code> - <code>*hspi</code>: SPI句柄 - <code>uint8_t *pData</code>: 发送数据缓冲区 - <code>uint16_t Size</code>: 发送数据大小
<code>HAL_SPI_Receive_IT</code>	中断模式接收数据	<ul style="list-style-type: none"> - <code>SPI_HandleTypeDef</code> - <code>*hspi</code>: SPI句柄 - <code>uint8_t *pData</code>: 接收数据缓冲区 - <code>uint16_t Size</code>: 接收数据大小
<code>HAL_SPI_TransmitReceive_IT</code>	中断模式发送和接收数据	<ul style="list-style-type: none"> - <code>SPI_HandleTypeDef</code> - <code>*hspi</code>: SPI句柄 - <code>uint8_t *pTxData</code>: 发送数据缓冲区 - <code>uint8_t *pRxData</code>: 接收数据缓冲区 - <code>uint16_t Size</code>: 数据大小
<code>HAL_SPI_Transmit_DMA</code>	DMA模式发送数据	<ul style="list-style-type: none"> - <code>SPI_HandleTypeDef</code> - <code>*hspi</code>: SPI句柄 - <code>uint8_t *pData</code>: 发送数据缓冲区 - <code>uint16_t Size</code>: 发送数据大小
<code>HAL_SPI_Receive_DMA</code>	DMA模式接收数据	<ul style="list-style-type: none"> - <code>SPI_HandleTypeDef</code> - <code>*hspi</code>: SPI句柄 - <code>uint8_t *pData</code>: 接收数据缓冲区 - <code>uint16_t Size</code>: 接收数据大小
<code>HAL_SPI_TransmitReceive_DMA</code>	DMA模式发送和接收数据	<ul style="list-style-type: none"> - <code>SPI_HandleTypeDef</code> - <code>*hspi</code>: SPI句柄 - <code>uint8_t *pTxData</code>: 发送数据缓冲区 - <code>uint8_t *pRxData</code>: 接收数据缓冲区 - <code>uint16_t Size</code>: 数据大小

3. 软件IIC

1. Cubemx配置

IIC需要上拉电阻

GPIO Mode and Configuration

Pinout view System view

Configuration

Group By Peripherals

GPIO Single Mapped Signals LPUART RCC SYS NVIC

Search Signals

Search (Ctrl+F) ☐ Show only Modified Pins

Pin	Signal	GPIO o	GPIO m	GPIO P	Maximu	Fast M	User La	Modified
PA5	n/a	Low	Output	No pull...	Low	n/a	LD2 [gr...	<input checked="" type="checkbox"/>
PB14	n/a	Low	Output	Pull-up	Low	n/a	SDA	<input checked="" type="checkbox"/>
PB15	n/a	Low	Output	Pull-up	Low	n/a	SCL	<input checked="" type="checkbox"/>
PC13	n/a	n/a	Externa...	No pull...	n/a	n/a	B1 [blu...	<input checked="" type="checkbox"/>

PB15 Configuration :

GPIO output level: Low

GPIO mode: Output Push Pull

GPIO Pull-up/Pull-down: Pull-up

Maximum output speed: Low

User Label: SCL

STM32G431RBTx LQFP64

Pinout view: VDD, VSS, PA12, PA11, PA10, PA9, PA8, PC9, PC8, PC7, PC6, PB15 (SCL), PB14 (SDA), PB13, PB12, PB11, PB10, PB9, PB8, PB7, PB6, PB5, PB4, PB3 (T_SWO), PC12, PC11, PC10, PA15, PA14 (T_SWCLK), PA13 (T_SWDIO)

2. 根据IIC通信协议进行程序编写，模拟通信过程