DMA：SPI

This tutorial demonstrates: **SPI (SPI2)** communication via **DMA**

1、software-hardware

- **STM32F103CubeIDE**
- **STM32 robot expansion board**

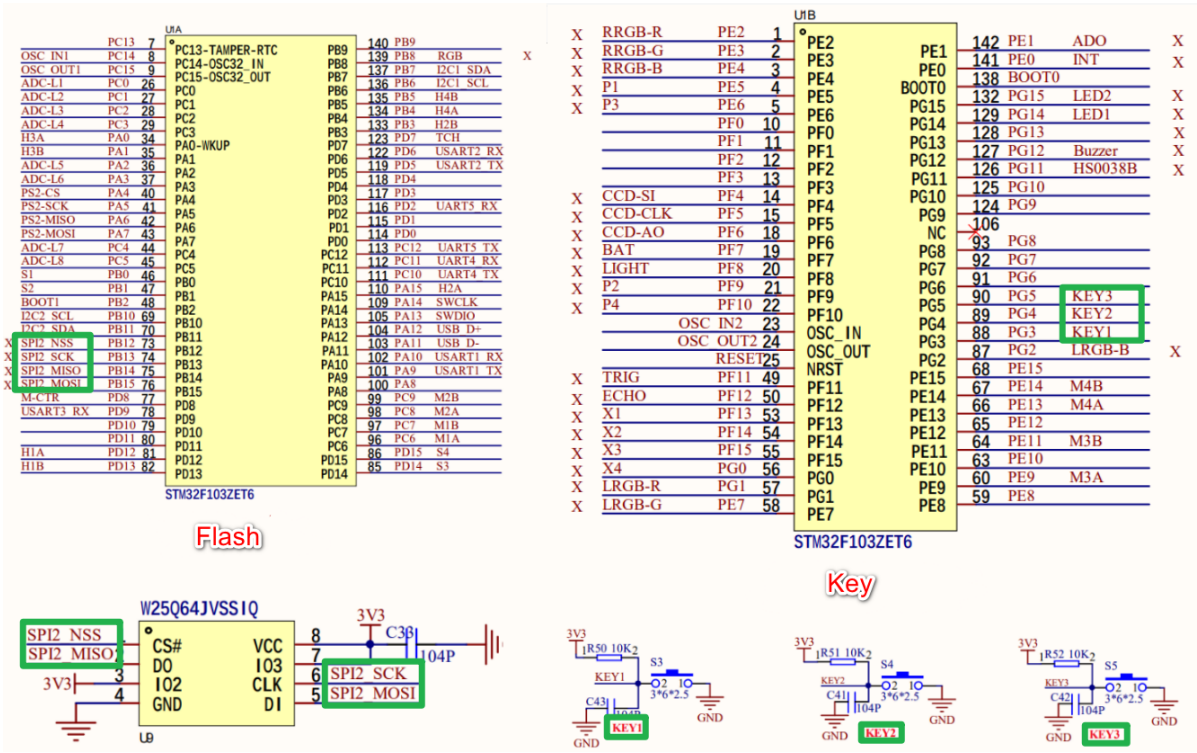
  USART, SPI, DMA: chip internal peripherals

  Key, external Flash (W25Q64) : onboard
- **Type-C cable or ST-Link**

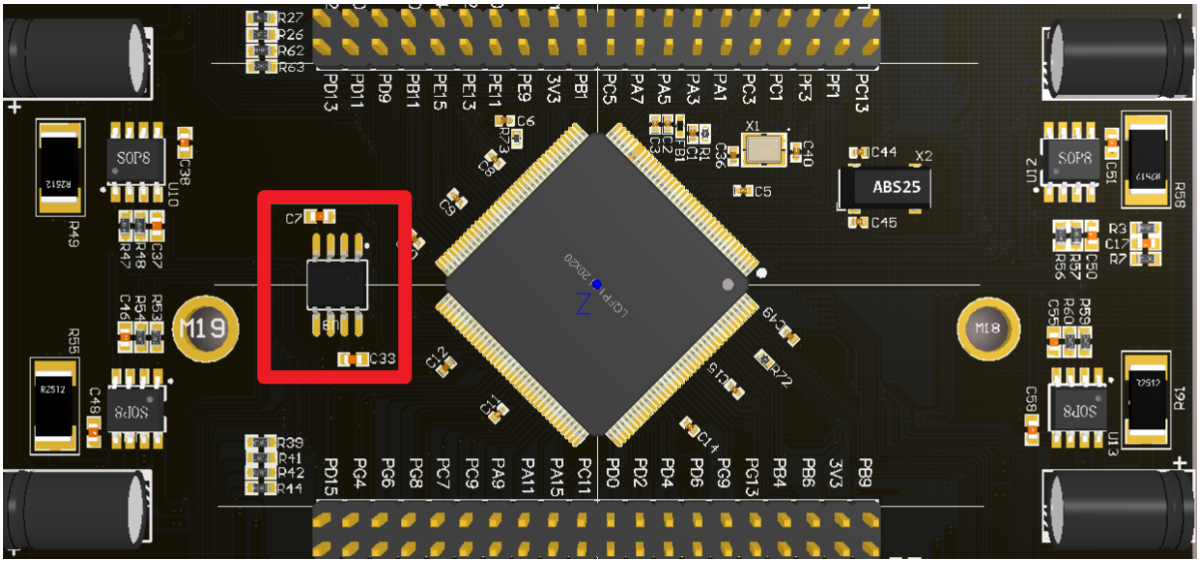
  Download or simulate the program of the development board

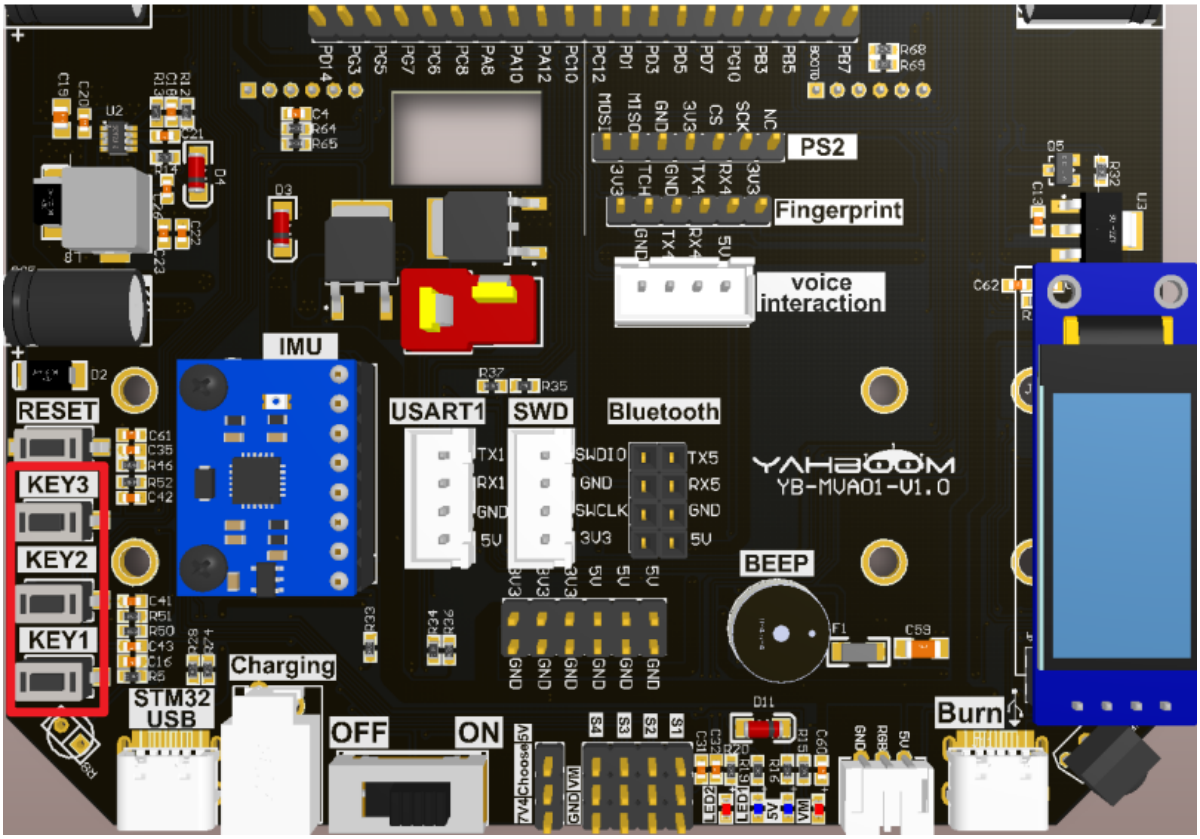
2、Brief principle

2.1、Hardware schematic diagram



2.2、Physical connection diagram





2.3、Principle of control

- **SPI**

Using SPI2 to communicate with external Flash (W25Q64), SPI related knowledge will not be introduced, you can see the previous [Chapter 3:3.14 Off-chip Flash] tutorial

- **DMA（Direct Memory Access）**

STM32F103ZET6 has a total of two DMA controllers, DMA1 has 7 channels, DMA2 has 5 channels;

It is used for high-speed data transfer between peripheral equipment and memory and between memory and memory.

DMA features

The initialization and start of DMA are completed by the CPU, and the transfer process is executed by the DMA controller without the participation of the CPU, so that the CPU resources are saved to do other operations.

DMA1 requests per channel

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
ADC1	ADC1	-	-	-	-	-	-
SPI/I^2S	-	SPI1_RX	SPI1_TX	SPI2/I2S2_RX	SPI2/I2S2_TX	-	-
USART	-	USART3_TX	USART3_RX	USART1_TX	USART1_RX	USART2_RX	USART2_TX
I^2C	-	-	-	I2C2_TX	I2C2_RX	I2C1_TX	I2C1_RX
TIM1	-	TIM1_CH1	-	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_UP	TIM1_CH3	-
TIM2	TIM2_CH3	TIM2_UP	-	-	TIM2_CH1	-	TIM2_CH2 TIM2_CH4
TIM3	-	TIM3_CH3	TIM3_CH4 TIM3_UP	-	-	TIM3_CH1 TIM3_TRIG	-
TIM4	TIM4_CH1	-	-	TIM4_CH2	TIM4_CH3	-	TIM4_UP

DMA2 requests per channel

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
ADC3(1)					ADC3
SPI/I2S3	SPI/I2S3_RX	SPI/I2S3_TX			
UART4			UART4_RX		UART4_TX
SDIO(1)				SDIO	
TIM5	TIM5_CH4 TIM5_TRIG	TIM5_CH3 TIM5_UP		TIM5_CH2	TIM5_CH1
TIM6/ DAC_Channel1			TIM6_UP/ DAC_Channel1		
TIM7				TIM7_UP/ DAC_Channel2	
TIM8	TIM8_CH3 TIM8_UP	TIM8_CH4 TIM8_TRIG TIM8_COM	TIM8_CH1		TIM8_CH2

DMA1 channels 4 and 5 are used in this tutorial

3、Engineering configuration

Project Configuration: Prompts for configuration options in the STM32CubeIDE project configuration process

3.1、Notes

Omitted project configuration: **New project, chip selection, project configuration, SYS for pin configuration, RCC configuration, clock configuration, and project configuration** content

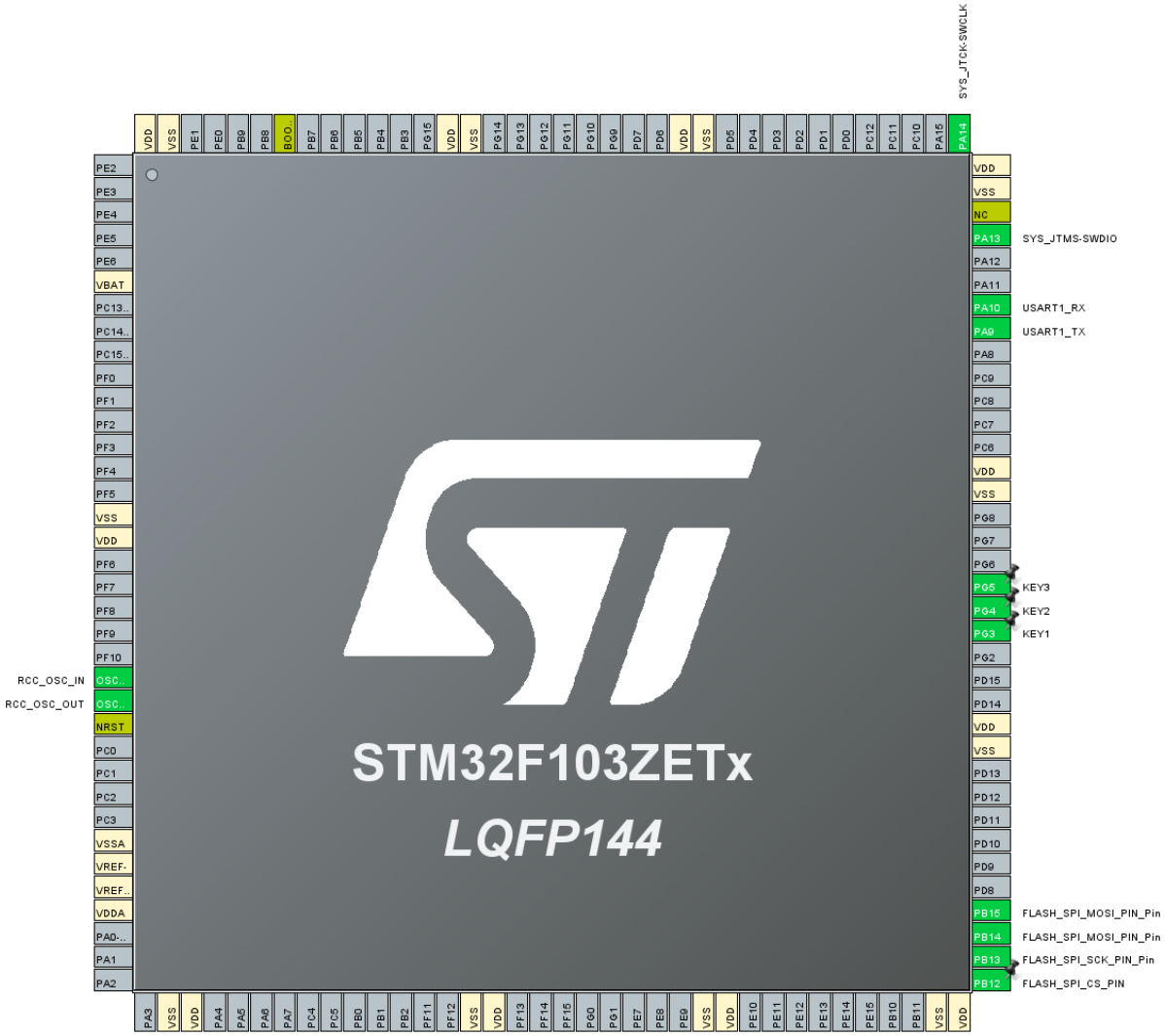
The project configuration part, which is not omitted, is the key point to configure in this tutorial.

```
Please refer to [2. Development environment construction and use: STM32CubeIDE
installation and use] to understand how to configure the omitted parts of the
project.
```

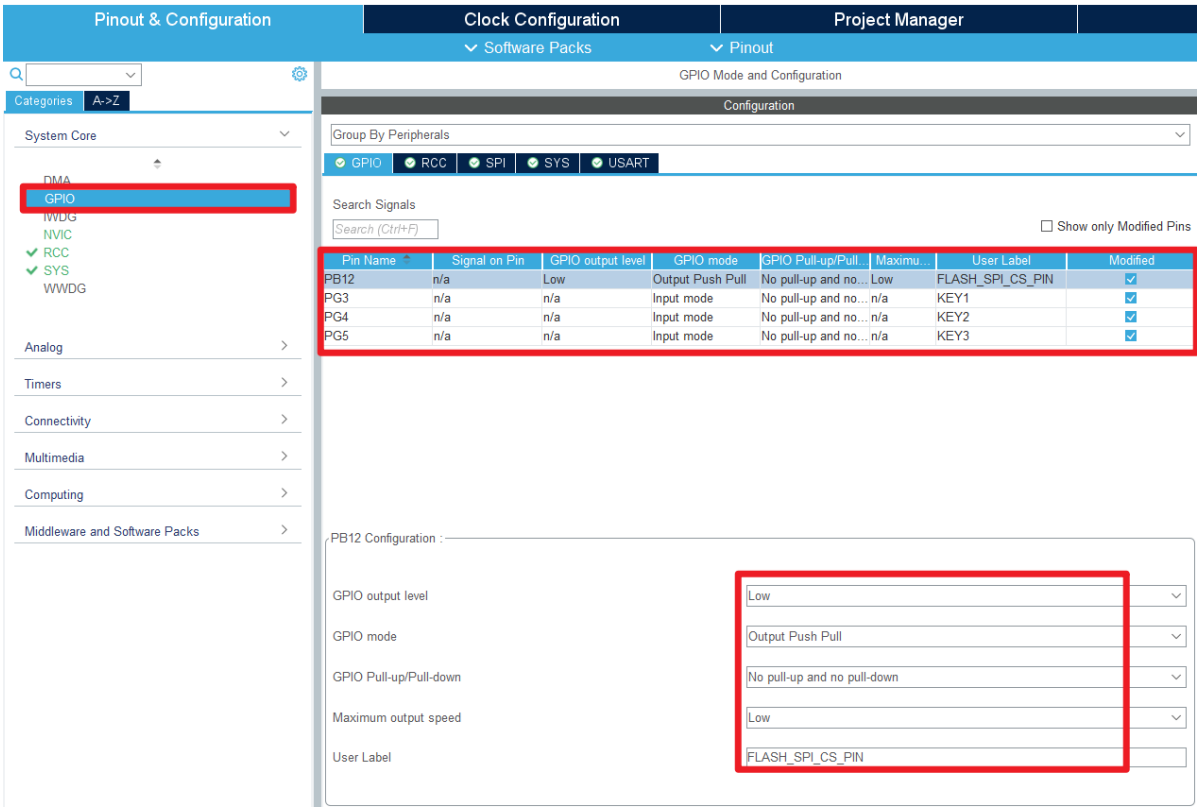
3.2、Pin configuration

- **Configure the specified pin function**

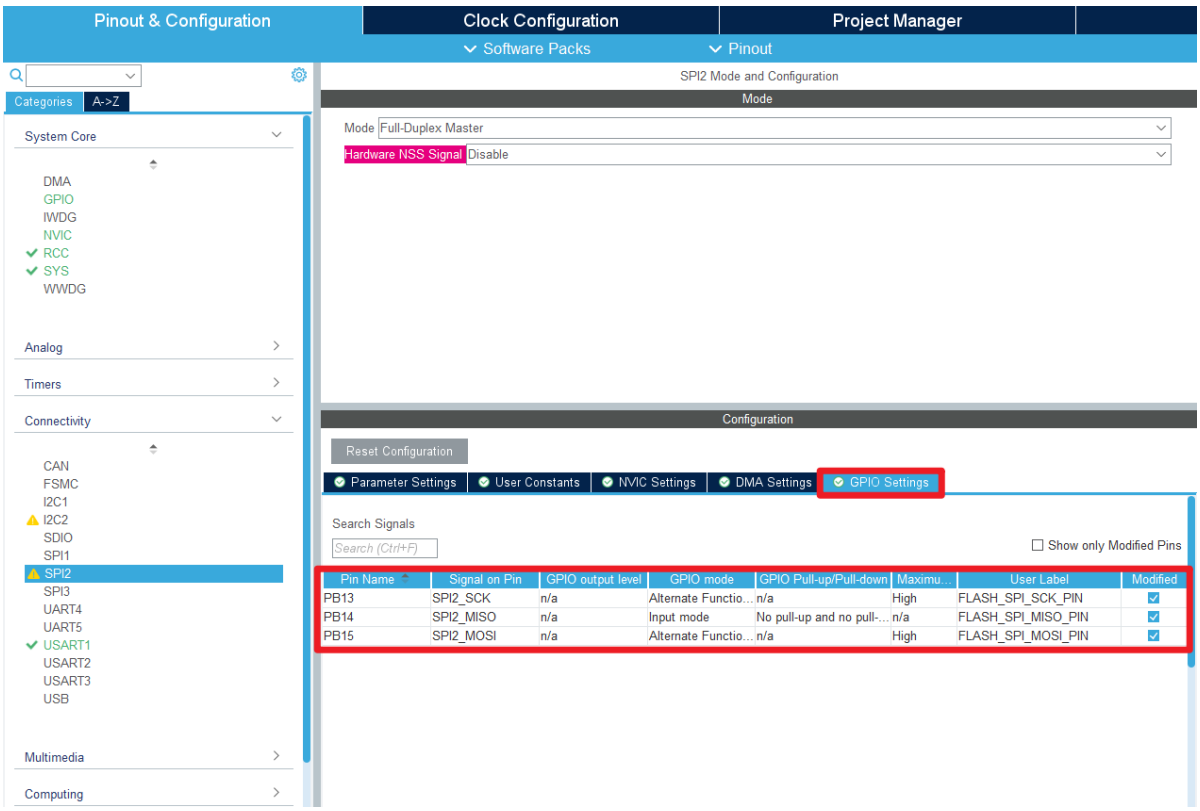
You can directly select the corresponding pin number in the pin view, and the corresponding option will appear when the mouse is left clicked

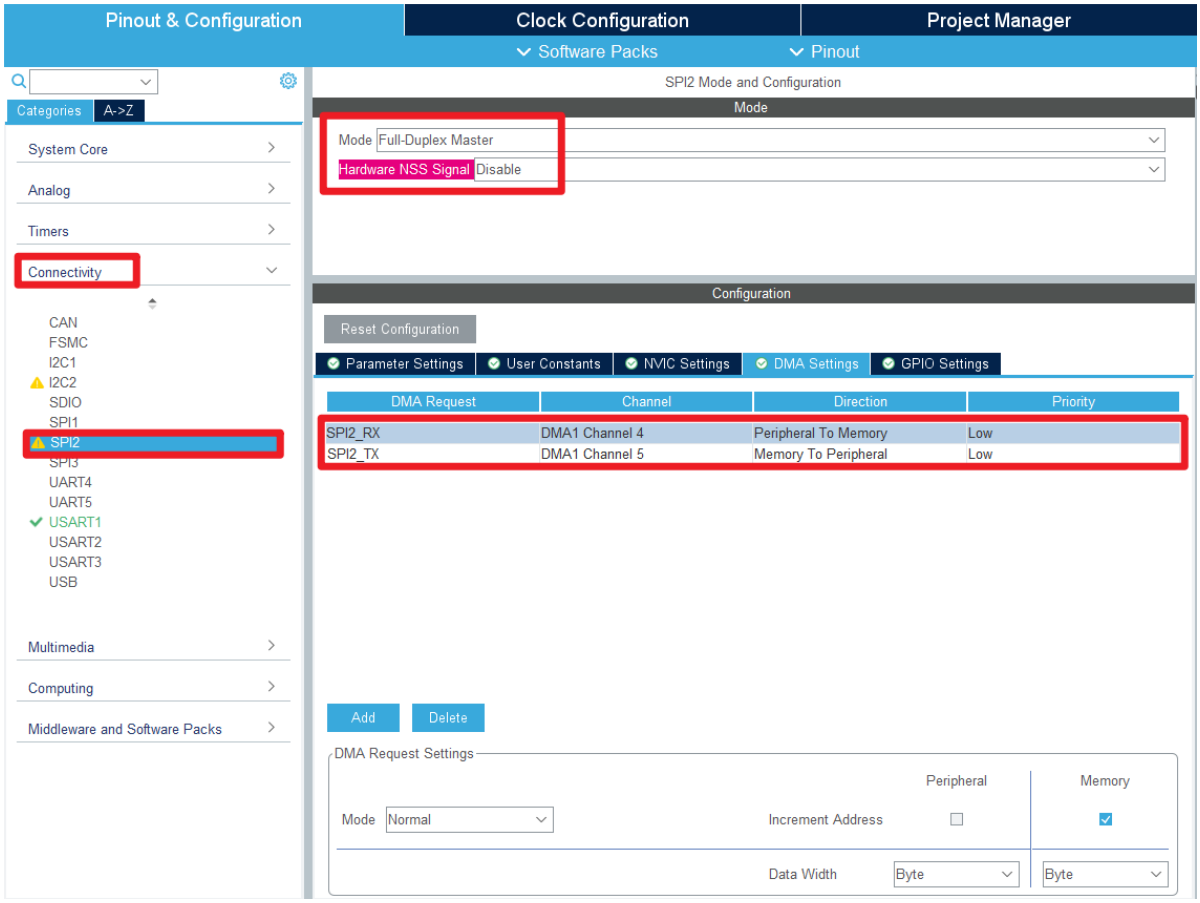


- **GPIO**

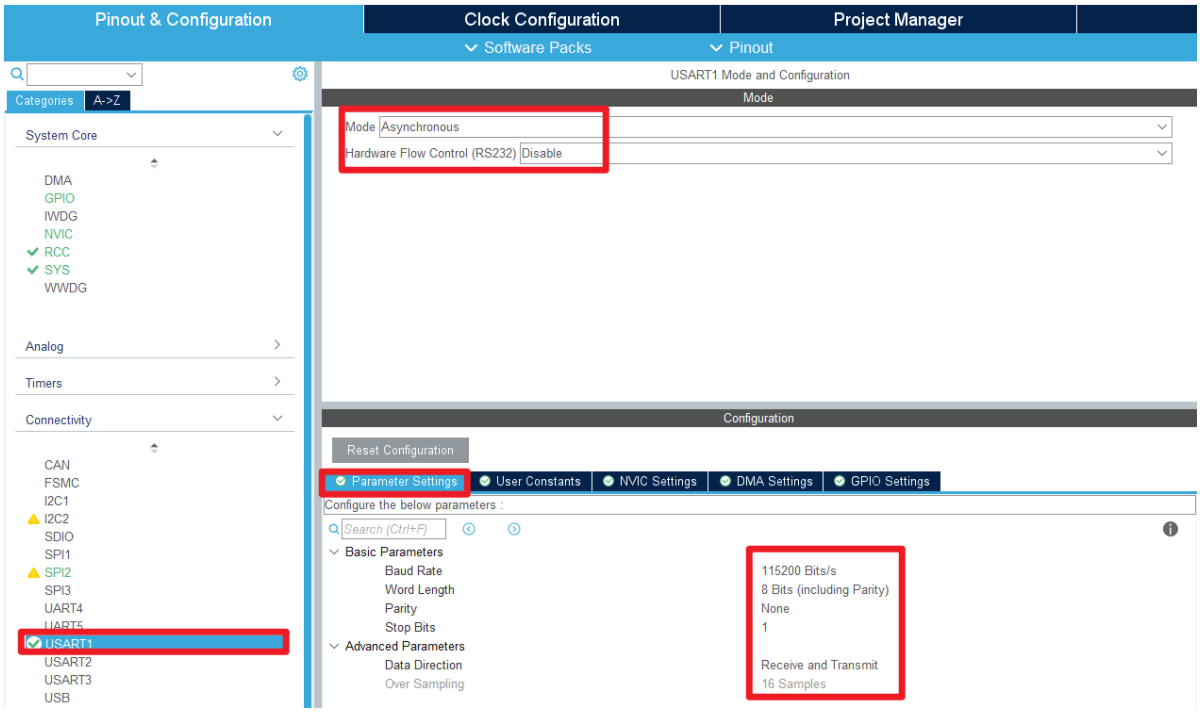

Pinout & Configuration
Clock Configuration
Project Manager
Software Packs
Pinout
Categories
A->Z
System Core
DMA
GPIO
IWDG
NVIC
RCC
SYS
WWDG
Analog
Timers
Connectivity
Multimedia
Computing
Middleware and Software Packs
GPIO Mode and Configuration
Configuration
Group By Peripherals
GPIO
RCC
SPI
SYS
USART
Search Signals
Search (Ctrl+F)
Show only Modified Pins
Pin Name
Signal on Pin
GPIO output level
GPIO mode
GPIO Pull-up/Pull...
Maximu...
User Label
Modified
PB12
n/a
Low
Output Push Pull
No pull-up and no...
Low
FLASH_SPI_CS_PIN
PG3
n/a
n/a
Input mode
No pull-up and no...
n/a
KEY1
PG4
n/a
n/a
Input mode
No pull-up and no...
n/a
KEY2
PG5
n/a
n/a
Input mode
No pull-up and no...
n/a
KEY3
PB12 Configuration :
GPIO output level
Low
GPIO mode
Output Push Pull
GPIO Pull-up/Pull-down
No pull-up and no pull-down
Maximum output speed
Low
User Label
FLASH_SPI_CS_PIN

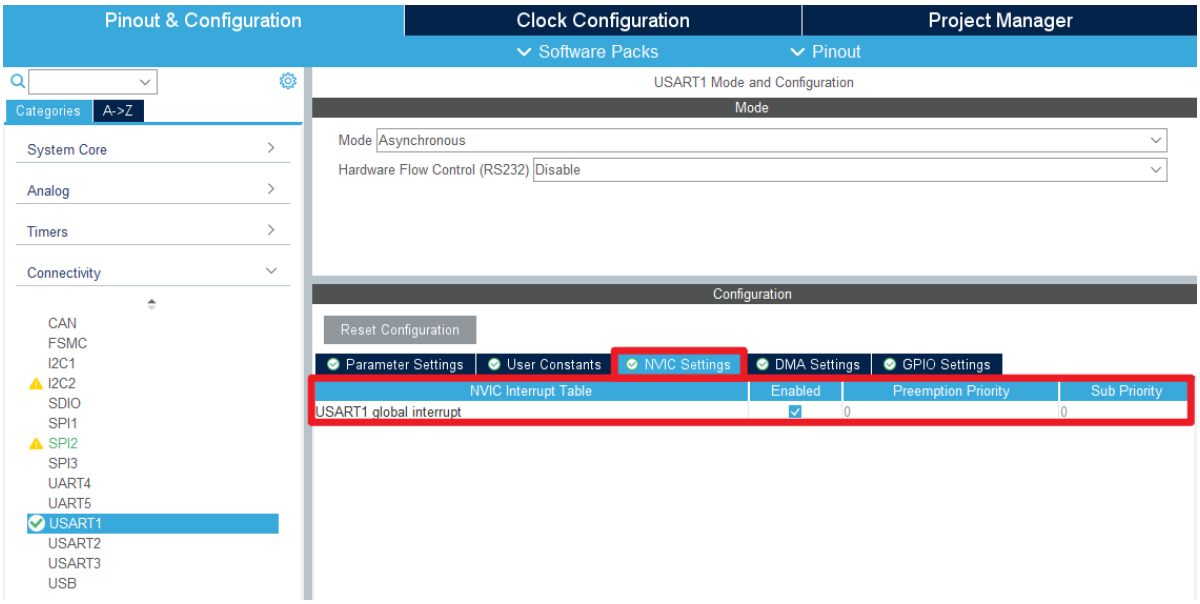
- **SPI**


Pinout & Configuration
Clock Configuration
Project Manager
Software Packs
Pinout
Categories
A->Z
System Core
DMA
GPIO
IWDG
NVIC
RCC
SYS
WWDG
Analog
Timers
Connectivity
CAN
FSMC
I2C1
I2C2
SDIO
SPI1
SPI2
SPI3
UART4
UART5
USART1
USART2
USART3
USB
Multimedia
Computing
SPI2 Mode and Configuration
Mode
Mode
Full-Duplex Master
Hardware NSS Signal
Disable
Configuration
Reset Configuration
Parameter Settings
User Constants
NVIC Settings
DMA Settings
GPIO Settings
Search Signals
Search (Ctrl+F)
Show only Modified Pins
Pin Name
Signal on Pin
GPIO output level
GPIO mode
GPIO Pull-up/Pull-down
Maximu...
User Label
Modified
PB13
SPI2_SCK
n/a
Alternate Functio...
n/a
High
FLASH_SPI_SCK_PIN
PB14
SPI2_MISO
n/a
Input mode
No pull-up and no pull-...
n/a
FLASH_SPI_MISO_PIN
PB15
SPI2_MOSI
n/a
Alternate Functio...
n/a
High
FLASH_SPI_MOSI_PIN

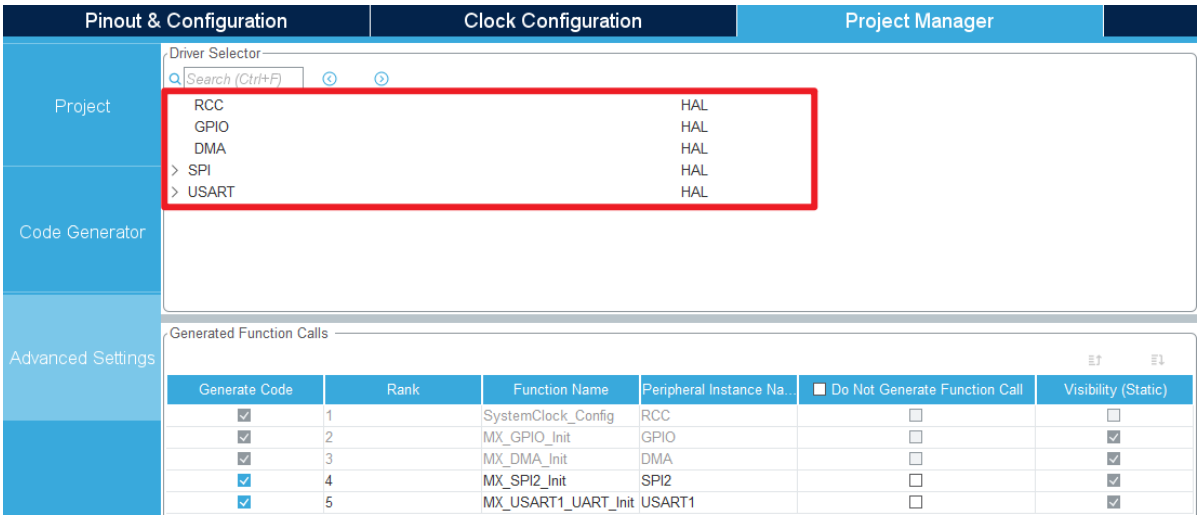

Pinout & Configuration
Clock Configuration
Project Manager
Software Packs
Pinout
Categories
A->Z
System Core
Analog
Timers
Connectivity
CAN
FSMC
I2C1
I2C2
SDIO
SPI1
SPI2
SPI3
UART4
UART5
USART1
USART2
USART3
USB
Multimedia
Computing
Middleware and Software Packs
SPI2 Mode and Configuration
Mode
Mode Full-Duplex Master
Hardware NSS Signal Disable
Configuration
Reset Configuration
Parameter Settings
User Constants
NVIC Settings
DMA Settings
GPIO Settings
DMA Request
Channel
Direction
Priority
SPI2_RX
DMA1 Channel 4
Peripheral To Memory
Low
SPI2_TX
DMA1 Channel 5
Memory To Peripheral
Low
Add
Delete
DMA Request Settings
Peripheral
Memory
Mode Normal
Increment Address
Data Width
Byte
Byte

- **USART**

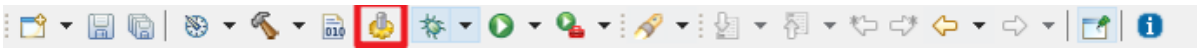

Pinout & Configuration
Clock Configuration
Project Manager
Software Packs
Pinout
Categories
A->Z
System Core
DMA
GPIO
IWDG
NVIC
RCC
SYS
WWDG
Analog
Timers
Connectivity
CAN
FSMC
I2C1
I2C2
SDIO
SPI1
SPI2
SPI3
UART4
UART5
USART1
USART2
USART3
USB
USART1 Mode and Configuration
Mode
Mode Asynchronous
Hardware Flow Control (RS232) Disable
Configuration
Reset Configuration
Parameter Settings
User Constants
NVIC Settings
DMA Settings
GPIO Settings
Configure the below parameters :
Search (Ctrl+F)
Basic Parameters
Baud Rate
Word Length
Parity
Stop Bits
Advanced Parameters
Data Direction
Over Sampling
115200 Bits/s
8 Bits (including Parity)
None
1
Receive and Transmit
16 Samples



- **Advanced Settings**



- **Generating code**



4、Main function

This section mainly introduces the functional code written by users. **Detailed code can be opened by yourself in the project file we provide, and enter the Bsp folder to view the source code.** .

4.1、User function

[Chapter 3: Off-chip Flash] Functions introduced are no longer covered

> **function：SPI_Read_Write_Byte**

Function prototypes	**uint8_t SPI_Read_Write_Byte(uint8_t BYTE)**
Functional Description	1 byte of data is sent via SPI
Input parameters	**BYTE**：data

Function prototypes	**uint8_t SPI_Read_Write_Byte(uint8_t BYTE)**
Return value	0
Tips	defined DMA_SPI_USE macros used to switch the DMA data transmission or

> **function：SPI_Write_DMA**

Function prototypes	**uint16_t SPI_Write_DMA(uint8_t *pTxData, uint16_t Size)**
Functional Description	Data is sent via DMA
Input parameters1	**pTxData**：Data head address
Input parameters2	**Size**：Number of bytes of data
Return value	**HAL status value**：HAL_OK、HAL_ERROR、HAL_BUSY、HAL_TIMEOUT（0-3）

4.2、HAL library function parsing

The HAL library functions that were covered in the previous tutorial will not be covered

```
If you want to find the HAL library and LL library function analysis involved in the entire tutorial, you can view the documents in the folder [8. STM32 Manual: STM32F1_HAL Library and LL Library_User Manual]
```

> **function：HAL_DMA_Init**

Function prototypes	**HAL_StatusTypeDef HAL_DMA_Init(DMA_HandleTypeDef *hdma)**
Functional Description	Initialize the DMA controller
Input parameters	**hdma**：DMA handle address
Return value	**HAL status value**：HAL_OK、HAL_ERROR、HAL_BUSY、HAL_TIMEOUT

> **function：HAL_SPI_TransmitReceive_DMA**

Function prototypes	**HAL_StatusTypeDef HAL_SPI_TransmitReceive_DMA (SPI_HandleTypeDef *hspi, uint8_t *pTxData, uint8_t *pRxData,uint16_t Size)**
Functional Description	Sending and receiving SPI data via DMA mode

Function prototypes	**HAL_StatusTypeDef HAL_SPI_TransmitReceive_DMA (SPI_HandleTypeDef *hspi, uint8_t *pTxData, uint8_t *pRxData,uint16_t Size)**
Input parameters1	**hspi**：SPI handle address
Input parameters2	**pTxData**：The first address to send data to
Input parameters3	**pRxData**：The first address to receive data
Input parameters4	**Size**：The number of bytes of data sent and received
Return value	**HAL status value**：HAL_OK、HAL_ERROR、HAL_BUSY、HAL_TIMEOUT

> **function：HAL_SPI_Transmit_DMA**

Function prototypes	**HAL_StatusTypeDef HAL_SPI_Transmit_DMA (SPI_HandleTypeDef *hspi, uint8_t *pData, uint16_t Size)**
Functional Description	SPI send function that transfers data using DMA
Input parameters1	**hspi**：SPI handle address
Input parameters2	**pData**：The first address of the data
Input parameters3	**Size**：Number of bytes of data
Return value	**HAL status value**：HAL_OK、HAL_ERROR、HAL_BUSY、HAL_TIMEOUT

5、Experimental phenomenon

After downloading the program successfully, press the RESET button of the development board to open the serial debugging assistant to observe the phenomenon

```
For program download, please refer to [2. Development environment construction and use: program download and simulation]
```

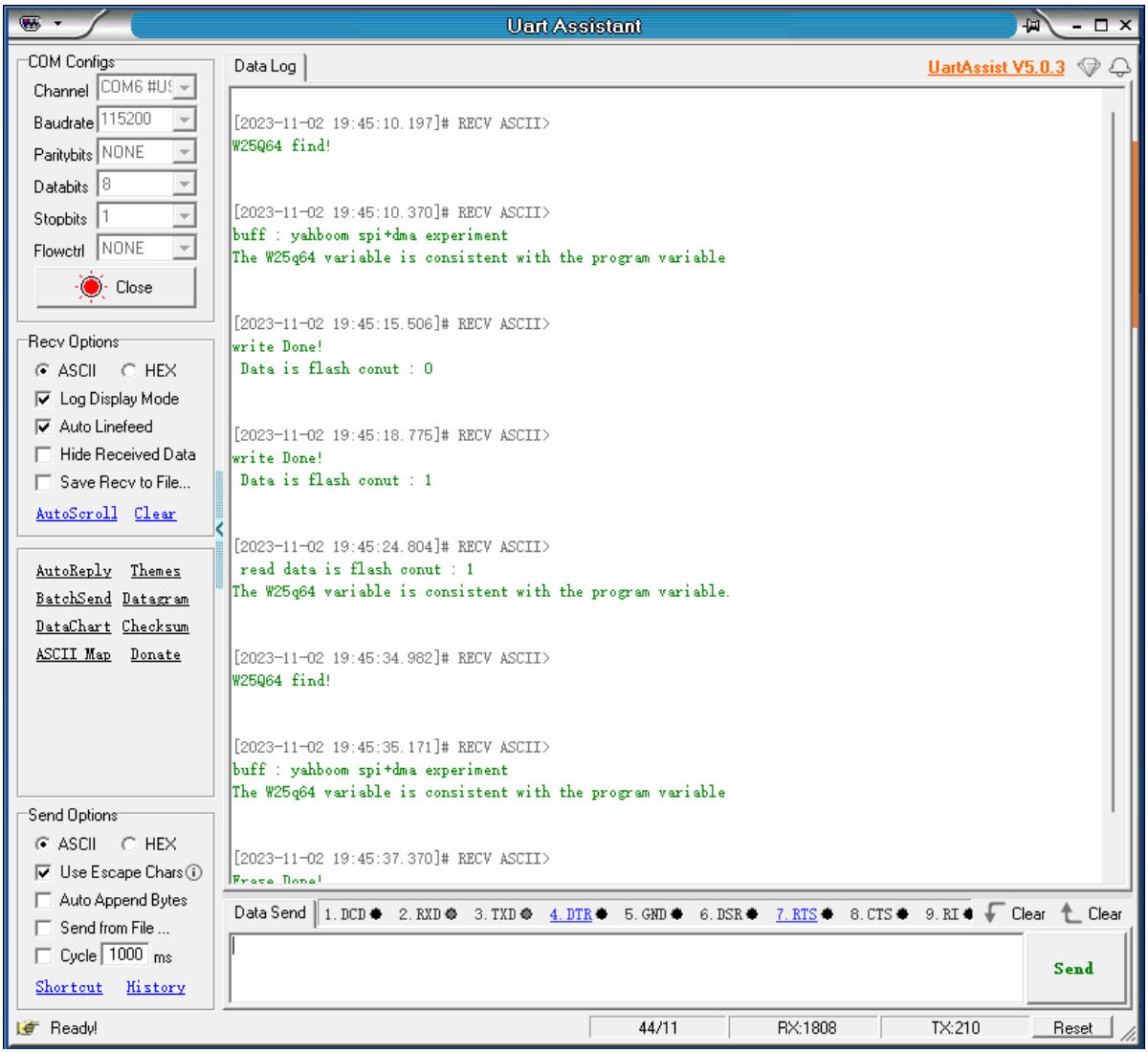
phenomenon：

Press RESET button: the serial debugging assistant will print the detection status of the external Flash chip;

Press KEY1: The serial debugging assistant will add 1 to the count value and write the value to the external Flash chip;

Press KEY2: The serial debugging assistant will output the numerical value stored in the specified sector of the external Flash chip;

Press KEY3: The serial debugging assistant will erase the specified sector data.


Uart Assistant
COM Configs
Channel COM6 #US
Baudrate 115200
Paritybits NONE
Databits 8
Stopbits 1
Flowctrl NONE
Close
Recv Options
ASCII HEX
Log Display Mode
Auto Linefeed
Hide Received Data
Save Recv to File...
AutoScroll Clear
AutoReply Themes
BatchSend Datagram
DataChart Checksum
ASCII Map Donate
Send Options
ASCII HEX
Use Escape Chars
Auto Append Bytes
Send from File ...
Cycle 1000 ms
Shortcut History
Data Log
UartAssist V5.0.3
[2023-11-02 19:45:10.197]# RECV ASCII>
W25Q64 find!
[2023-11-02 19:45:10.370]# RECV ASCII>
buff : yahboom spi+dma experiment
The W25q64 variable is consistent with the program variable
[2023-11-02 19:45:15.506]# RECV ASCII>
write Done!
Data is flash conut : 0
[2023-11-02 19:45:18.775]# RECV ASCII>
write Done!
Data is flash conut : 1
[2023-11-02 19:45:24.804]# RECV ASCII>
read data is flash conut : 1
The W25q64 variable is consistent with the program variable.
[2023-11-02 19:45:34.982]# RECV ASCII>
W25Q64 find!
[2023-11-02 19:45:35.171]# RECV ASCII>
buff : yahboom spi+dma experiment
The W25q64 variable is consistent with the program variable
[2023-11-02 19:45:37.370]# RECV ASCII>
Data Send
1. DCD 2. RXD 3. TXD 4. DTR 5. GND 6. DSR 7. RTS 8. CTS 9. RI
Clear Clear
Send
Ready!
44/11
RX:1808
TX:210
Reset