DMA： I2C

This tutorial demonstrates **I2C (I2C1)** communication via **DMA**

1、software-hardware

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

  I2C: chip internal peripheral

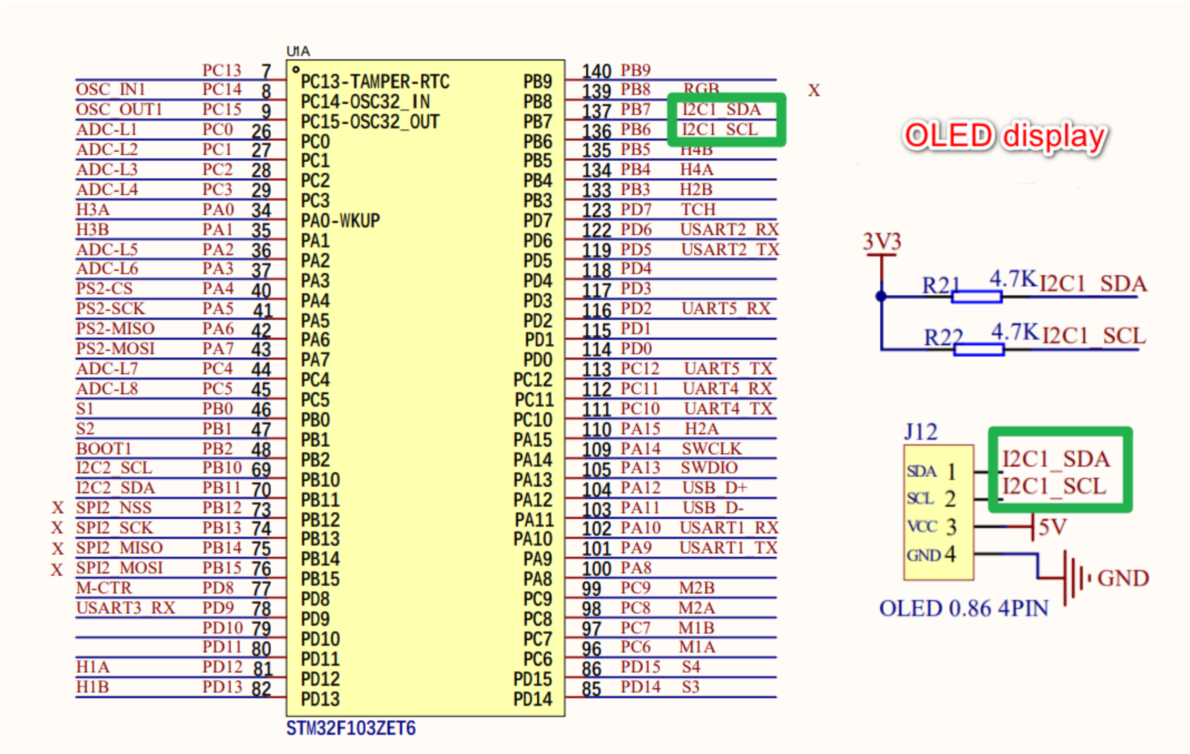
  OLED: External
- **Type-C cable or ST-Link**

  Download or simulate the program of the development board

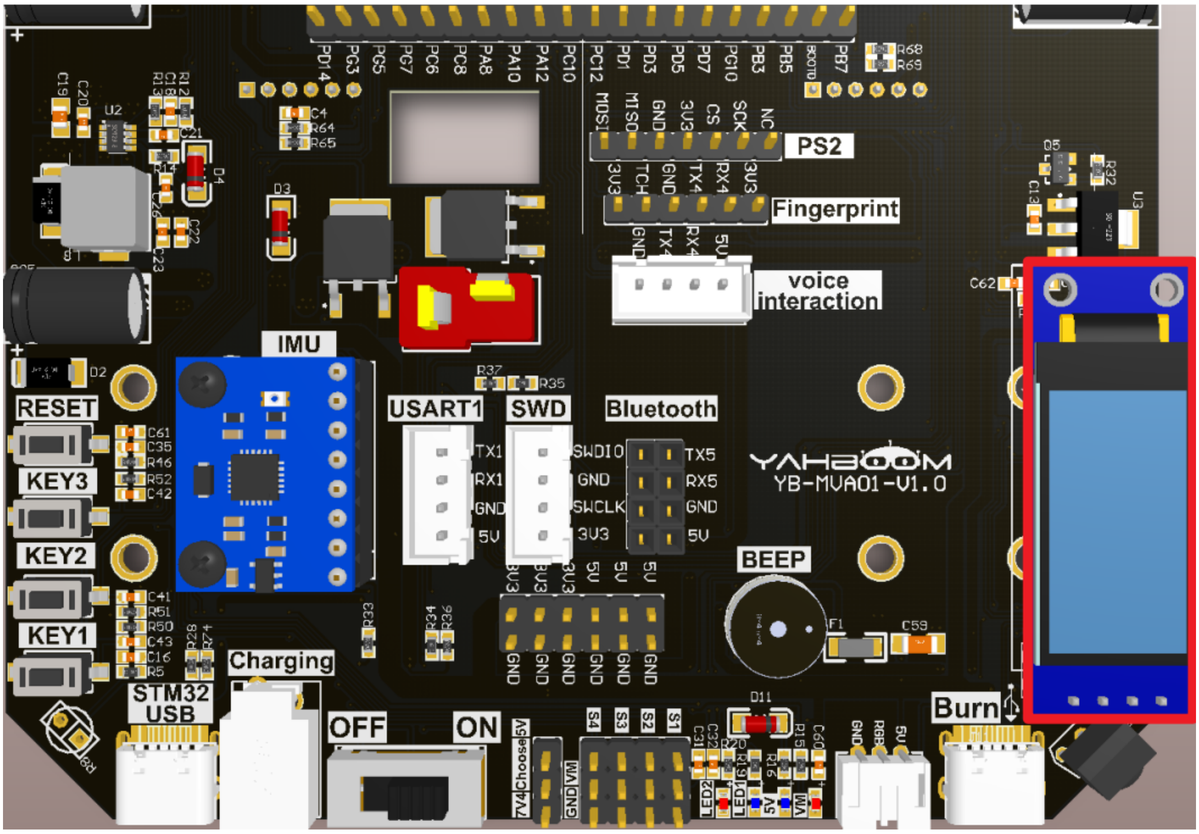
2、Brief principle

2.1、Hardware schematic diagram

The schematic shows only the I2C (I2C1) interface used in the tutorial



2.2、Physical connection diagram



OLED	STM32 board
VCC	VCC
SCL	SCL
SDA	SDA
GND	GND

2.3、Principle of control

- **I2C**

The use of I2C1 peripheral interface and 0.91 inch OLED display for communication, I2C related knowledge will not be introduced, you can see before [Chapter 3:3.12 I2C communication]

- **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 for each channel

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
ADC1	ADC1	-	-	-	-	-	-
SPI/I^{2}S	-	SPI1_RX	SPI1_TX	SPI2/I2S2_RX	SPI2/I2S2_TX	-	-
USART	-	USART3_TX	USART3_RX	USART1_TX	USART1_RX	USART2_RX	USART2_TX
I^{2}C	-	-	-	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 for each 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 channel 6 is 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

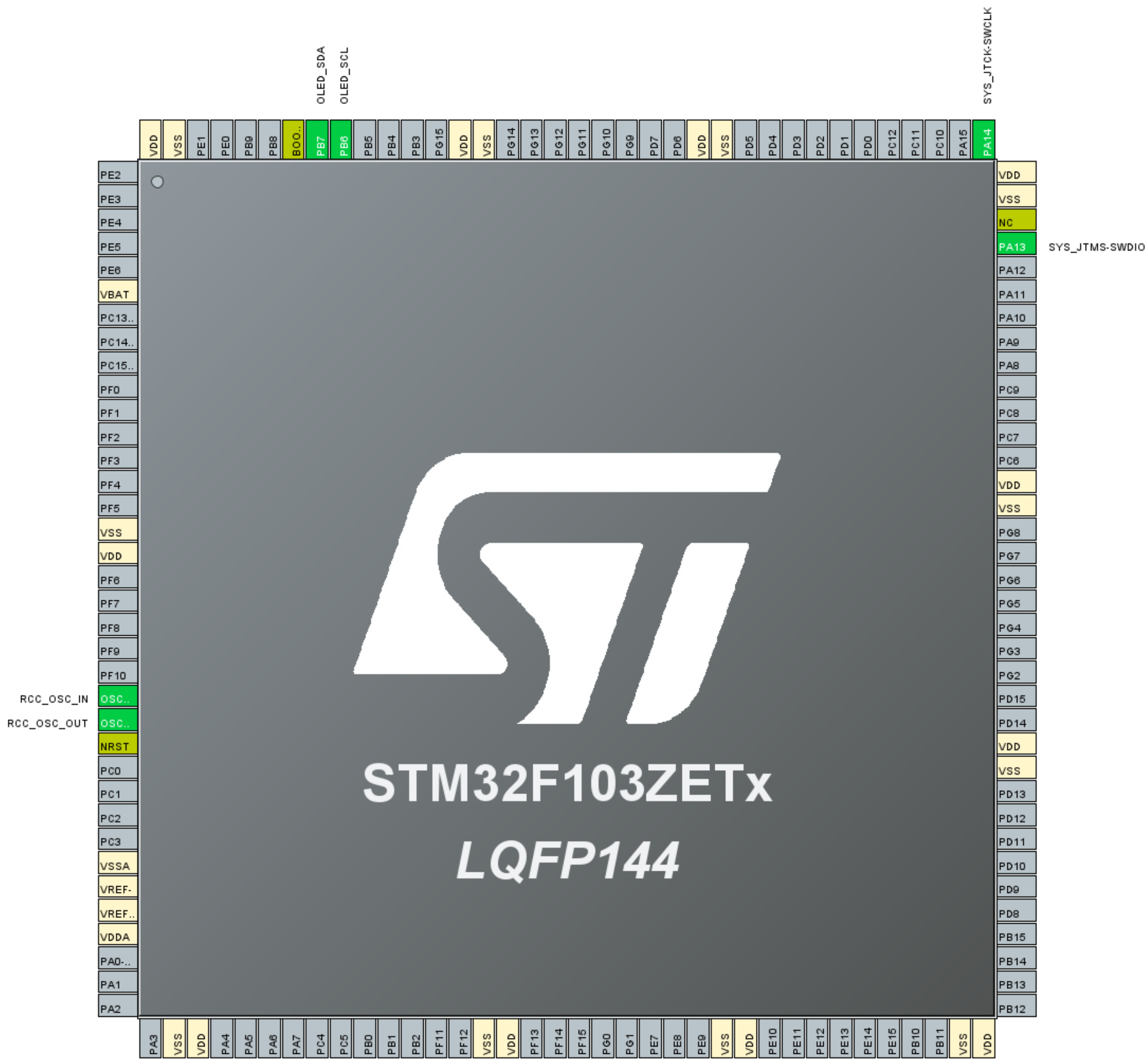
The project configuration part, which is not omitted, is the key point to configure in this tutorial.

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

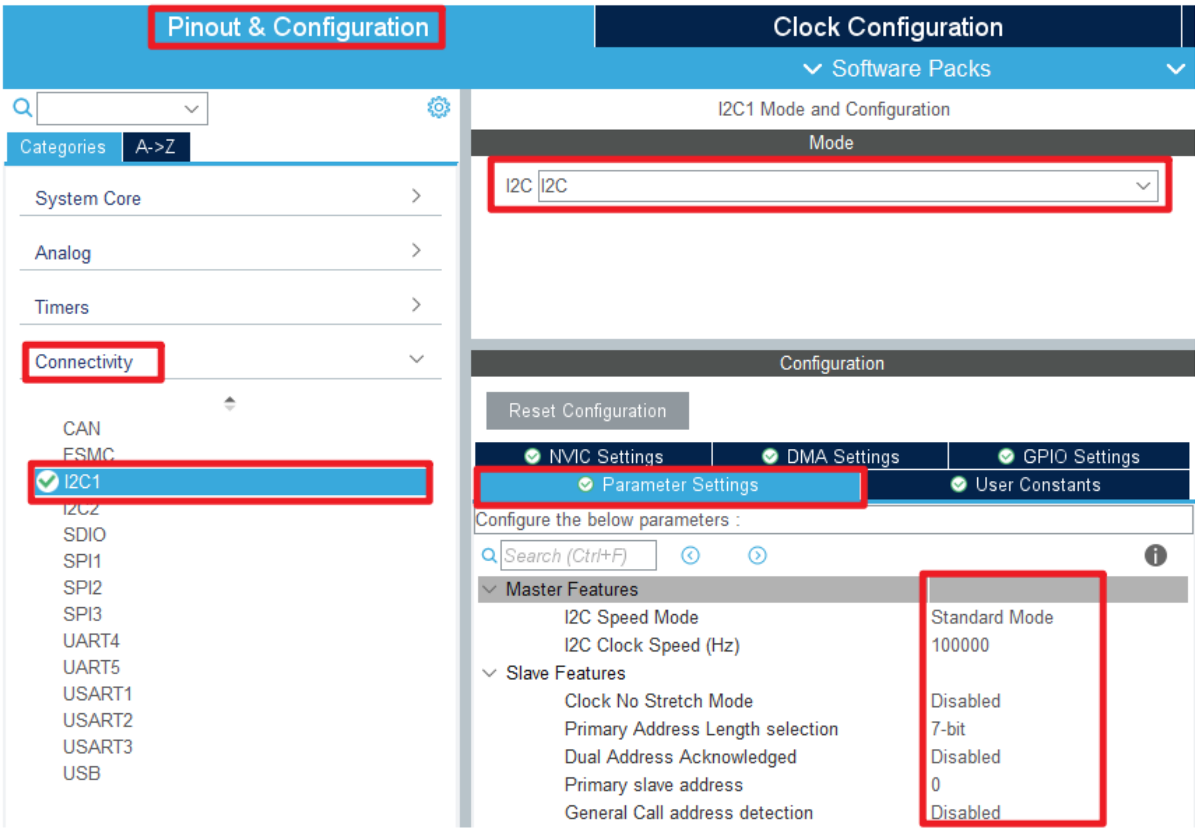
3.2、Pin configuration

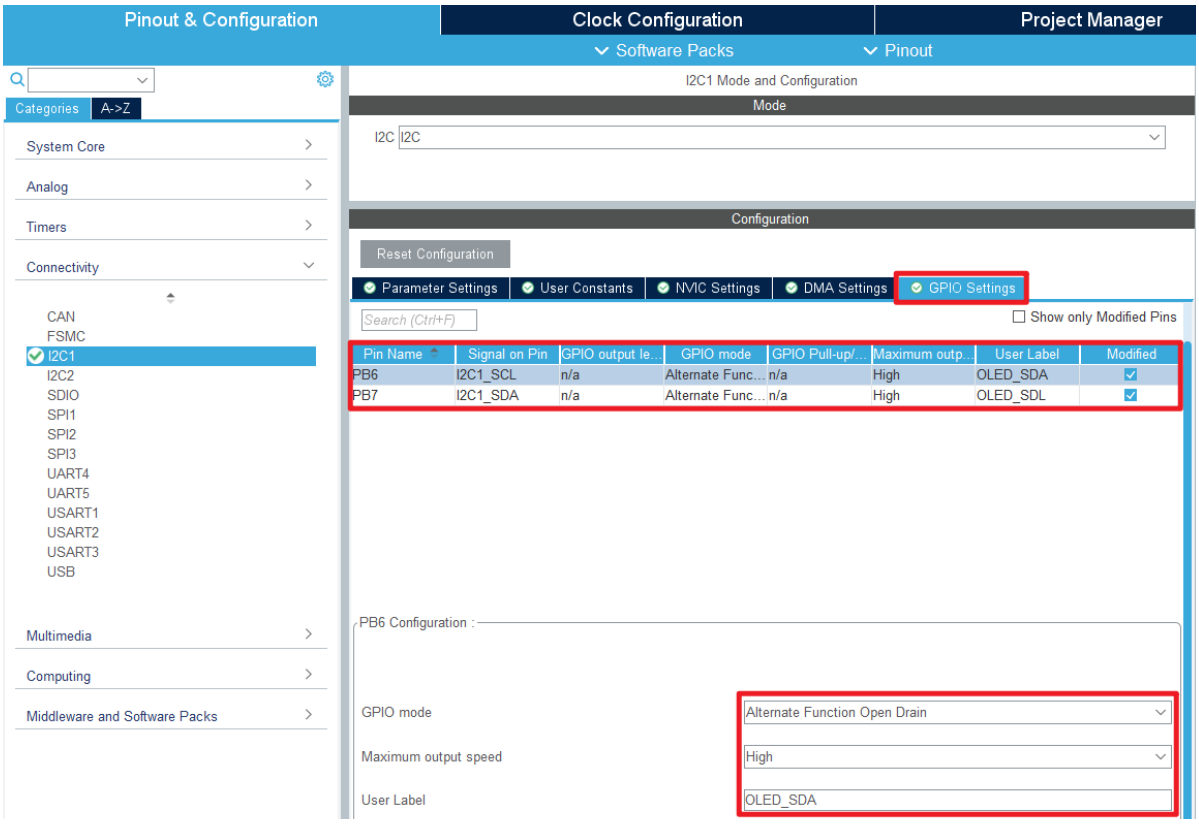
- **Configure the specified pin function**

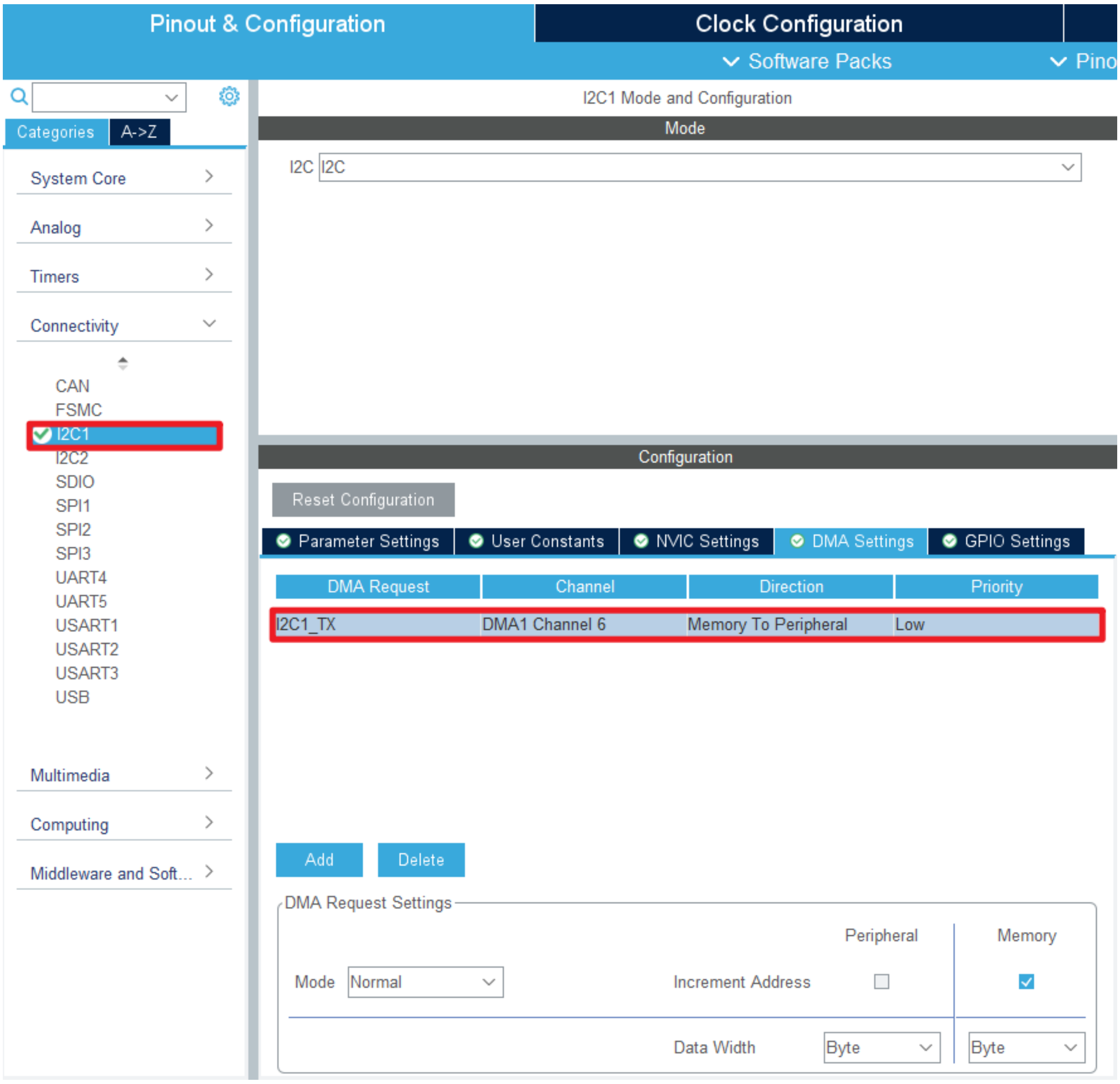
You can directly select the corresponding pin number in the pin view, and the corresponding option will appear when the mouse is left clicked

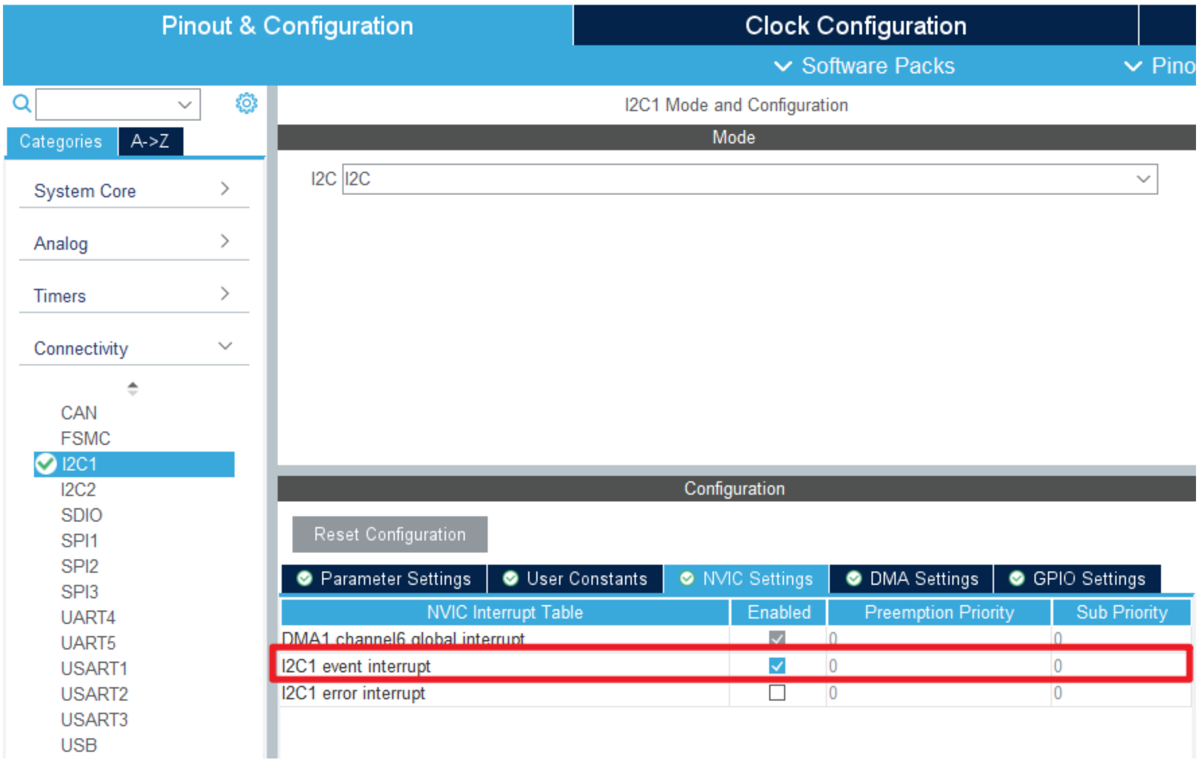


- **I2C**

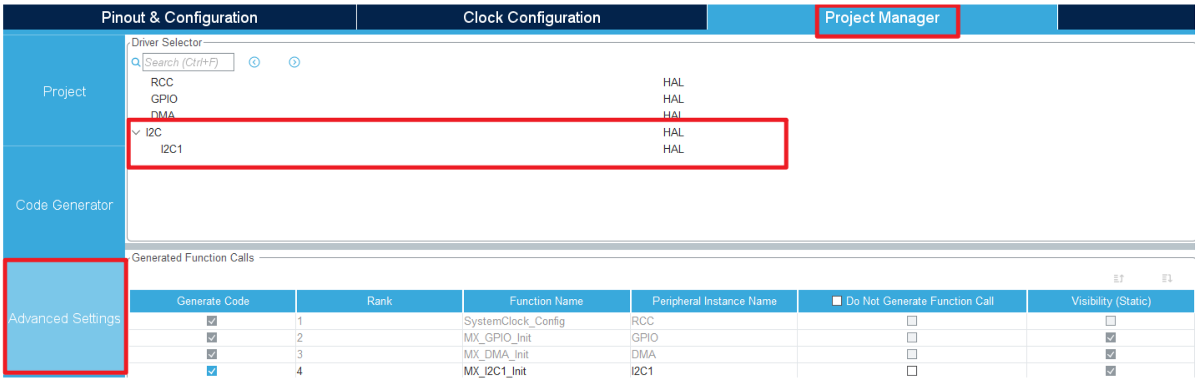

Pinout & Configuration
Clock Configuration
Software Packs
Categories
A->Z
System Core
Analog
Timers
Connectivity
CAN
FSMC
I2C1
I2C2
SDIO
SPI1
SPI2
SPI3
UART4
UART5
USART1
USART2
USART3
USB
I2C1 Mode and Configuration
Mode
I2C I2C
Configuration
Reset Configuration
NVIC Settings
DMA Settings
GPIO Settings
Parameter Settings
User Constants
Configure the below parameters :
Search (Ctrl+F)
Master Features
I2C Speed Mode Standard Mode
I2C Clock Speed (Hz) 100000
Slave Features
Clock No Stretch Mode Disabled
Primary Address Length selection 7-bit
Dual Address Acknowledged Disabled
Primary slave address 0
General Call address detection Disabled


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
I2C1 Mode and Configuration
Mode
I2C I2C
Configuration
Reset Configuration
Parameter Settings
User Constants
NVIC Settings
DMA Settings
GPIO Settings
Search (Ctrl+F)
Show only Modified Pins
Pin Name | Signal on Pin | GPIO output le... | GPIO mode | GPIO Pull-up/... | Maximum outp... | User Label | Modified
PB6 | I2C1_SCL | n/a | Alternate Func... | n/a | High | OLED_SDA
PB7 | I2C1_SDA | n/a | Alternate Func... | n/a | High | OLED_SDL
PB6 Configuration :
GPIO mode Alternate Function Open Drain
Maximum output speed High
User Label OLED_SDA

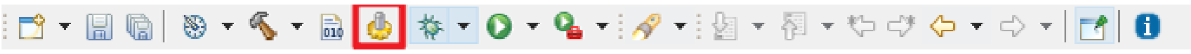




- **Advanced Settings**



- **生成代码**



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

Functions that have been covered in this tutorial will not be covered anymore.Feel free to refer to the previous tutorials

> **function：OLED_Draw_Line**

Function prototypes	**void OLED_Draw_Line(char *data, uint8_t line, bool clear, bool refresh)**
Functional Description	Write a single character
Input parameters1	**data**：Displaying data
Input parameters2	**line**：Number of rows
Input parameters3	**clear**：Clear content or ot
Input parameters4	**refresh**：refreash or not
Return value	None

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_I2C_Init**

Function prototypes	HAL_StatusTypeDef HAL_I2C_Init(I2C_HandleTypeDef *hi2c)
Functional Description	Initialize the I2C peripheral parameters
Input parameters	**hi2c**：I2C handle address
Return value	**HAL status value**：HAL_OK、HAL_ERROR、HAL_BUSY、HAL_TIMEOUT

> **function：HAL_I2C_MspInit**

Function prototypes	void HAL_I2C_MspInit(I2C_HandleTypeDef* i2cHandle)
Functional Description	**Initialize the clock, GPIO and NVIC of the I2C peripheral**
Input parameters	**i2cHandle**：I2C handle address
Return value	None

> **function：HAL_I2C_MspDeInit**

Function prototypes	void HAL_I2C_MspDeInit(I2C_HandleTypeDef* i2cHandle)
Functional Description	**Uninitialize the clock, GPIO, and NVIC of the I2C peripheral**
Input parameters	**i2cHandle**：I2C handle address
Return value	None

> **function：HAL_I2C_Mem_Write**

Function prototypes	HAL_StatusTypeDef HAL_I2C_Mem_Write(I2C_HandleTypeDef *hi2c, uint16_t DevAddress, uint16_t MemAddress, uint16_t MemAddSize, uint8_t *pData, uint16_t Size, uint32_t Timeout)
Functional Description	Write data to a specific memory address of the I2C device
Input parameters1	**hi2c**：I2C handle address
Input parameters2	**DevAddress**：Target device address
Input parameters3	**MemAddress**：Device memory address
Input parameters4	**MemAddSize**：The size of the device memory address
Input parameters5	**pData**：Data head address
Input parameters6	**Size**：Number of bytes of data
Input parameters7	**Timeout**：Timeout time

Function prototypes	**HAL_StatusTypeDef HAL_I2C_Mem_Write(I2C_HandleTypeDef *hi2c, uint16_t DevAddress, uint16_t MemAddress, uint16_t MemAddSize, uint8_t *pData, uint16_t Size, uint32_t Timeout)**
Return value	**HAL status value**： HAL_OK、HAL_ERROR、HAL_BUSY、HAL_TIMEOUT

> **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_I2C_Mem_Write_DMA**

Function prototypes	**HAL_StatusTypeDef HAL_I2C_Mem_Write_DMA(I2C_HandleTypeDef *hi2c, uint16_t DevAddress, uint16_t MemAddress, uint16_t MemAddSize, uint8_t *pData, uint16_t Size)**
Functional Description	Data is written to the internal registers of the I2C device by DMA
Input parameters1	**hi2c**： I2C handle address
Input parameters2	**DevAddress**： Target device address
Input parameters3	**MemAddress**： Device memory address
Input parameters4	**MemAddSize**： The size of the device memory address
Input parameters5	**pData**： Data head address
Input parameters6	**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 observe the OLED display

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

phenomenon:

OLED: The first line shows OLED init success! , the second line shows yahboom! , the third line shows This is a DMA+I2c!