Basic use of ultrasonic steering gear gimbal

This tutorial demonstrates: How to externally connect and use the ultrasonic servo gimbal on the expansion board and then print the ultrasonic ranging data through the serial port.

1. Software-Hardware

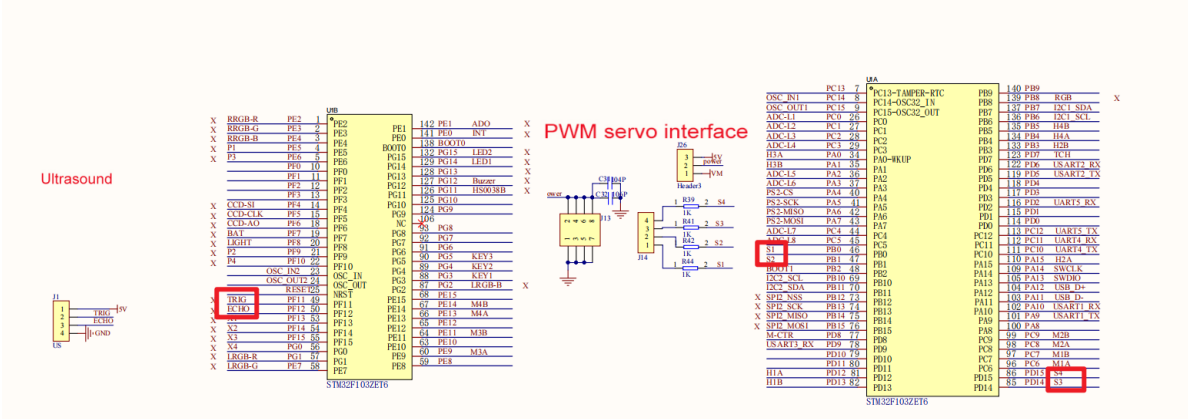
- **STM32F103CubeIDE**
- **STM32 expansion board**
- **Ultrasonic steering gear head**
- **Type-C data cable or ST-Link**

  Download programs or simulate the development board·
- **Serial Assistant**

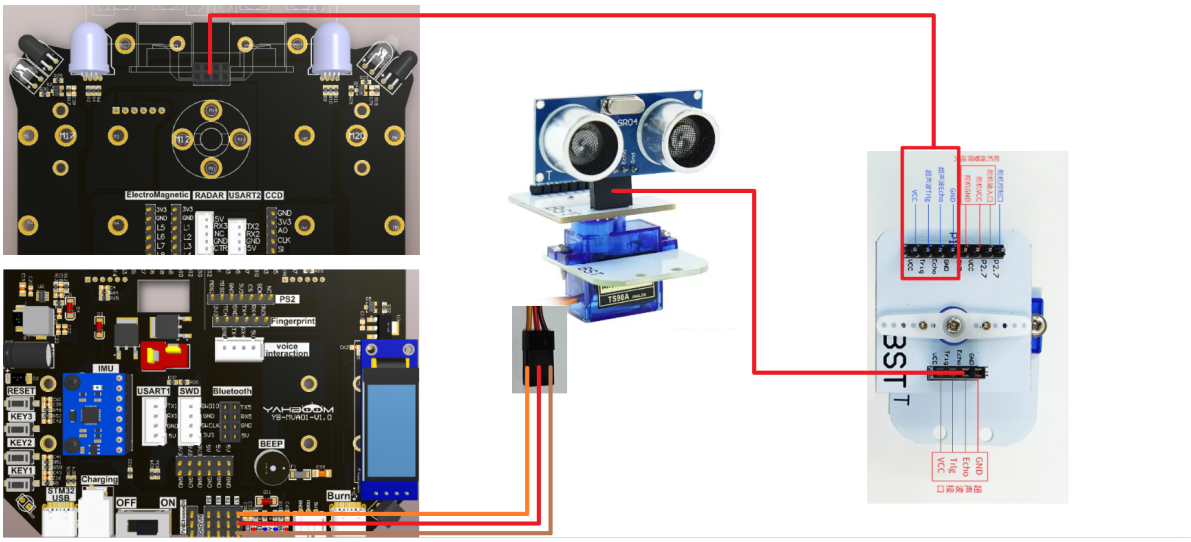
  Receive serial port data and print

2. Brief principle

1. Hardware schematic diagram



2、Physical connection diagram



3、Control principle

(Schematic name)	Control pin	Specific meaning
TRIG	PF11	Trigger terminal
ECHO	PF12	Receiver
S1	PB0	Servo control pin

Ultrasonic Module:

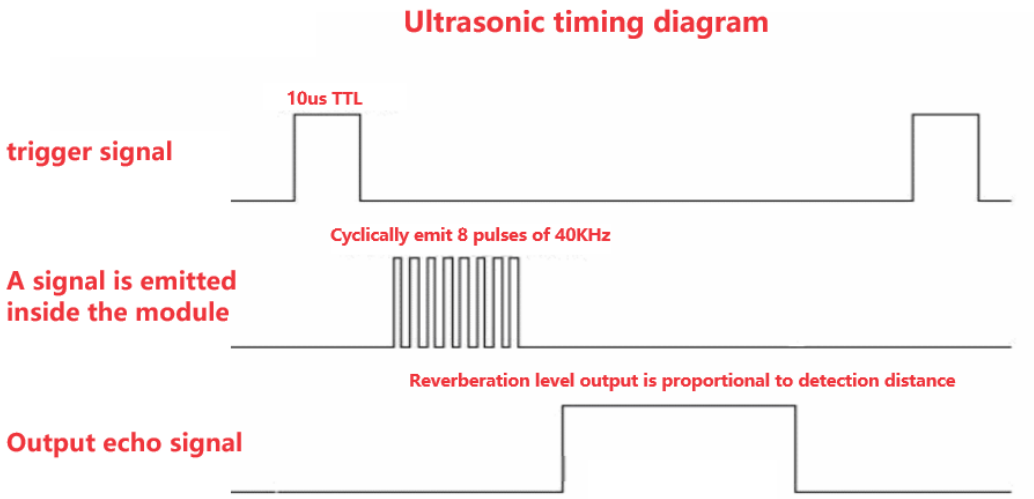
It is a module that uses ultrasonic waves for non-contact physical quantity measurement. It can accurately measure distance, speed, flow and other physical quantities by transmitting and receiving ultrasonic signals, and converts the measurement results into digital signal output. This article will popularize the ultrasonic module Its working principle and function.

Ultrasonic module information for this experiment:

Model	**HC-SR04**	**Detection distance**	**2-400cm**
Working voltage	5V	High precision	Up to 0.3cm
Operating current	15mA	Dead zone	2cm
Operating frequency	40KHz	Pin sequence	VCC, Trig (control end), Echo (receiving end), GND
Quiescent operating current	<2mA	Input trigger signal	10uS TTL pulse
Sensing angle	Not greater than 15°	Input echo signal	Output TTL level signal, proportional to range
Range range	2cm-4m (peak)	Level output	TTL level

Ranging principle: Input a high potential for more than 10 microseconds at the trigger end of the ultrasonic module to emit ultrasonic waves. After transmitting the ultrasonic waves and before receiving the returned ultrasonic waves, the receiving end is at a high potential. Therefore, the program can calculate the distance of the measured object from the high pulse duration of the "response" pin.

Test distance = (high level time * speed of sound (340M/S))/2;



Note: The above timing diagram shows that you only need to provide a pulse trigger signal of more than 10us, and the module will internally send out 8 40kHz cycle levels and detect echoes. Once an echo signal is detected, an echo signal is output. The pulse width of the reverberated signal is proportional to the measured distance. The distance can be calculated from the time interval between the transmitted signal and the received echo signal.

PWM control steering gear principle

The PWM signal is generated by using the timer/counter (TIM) peripheral, and the angle of the servo is controlled by adjusting the duty cycle.

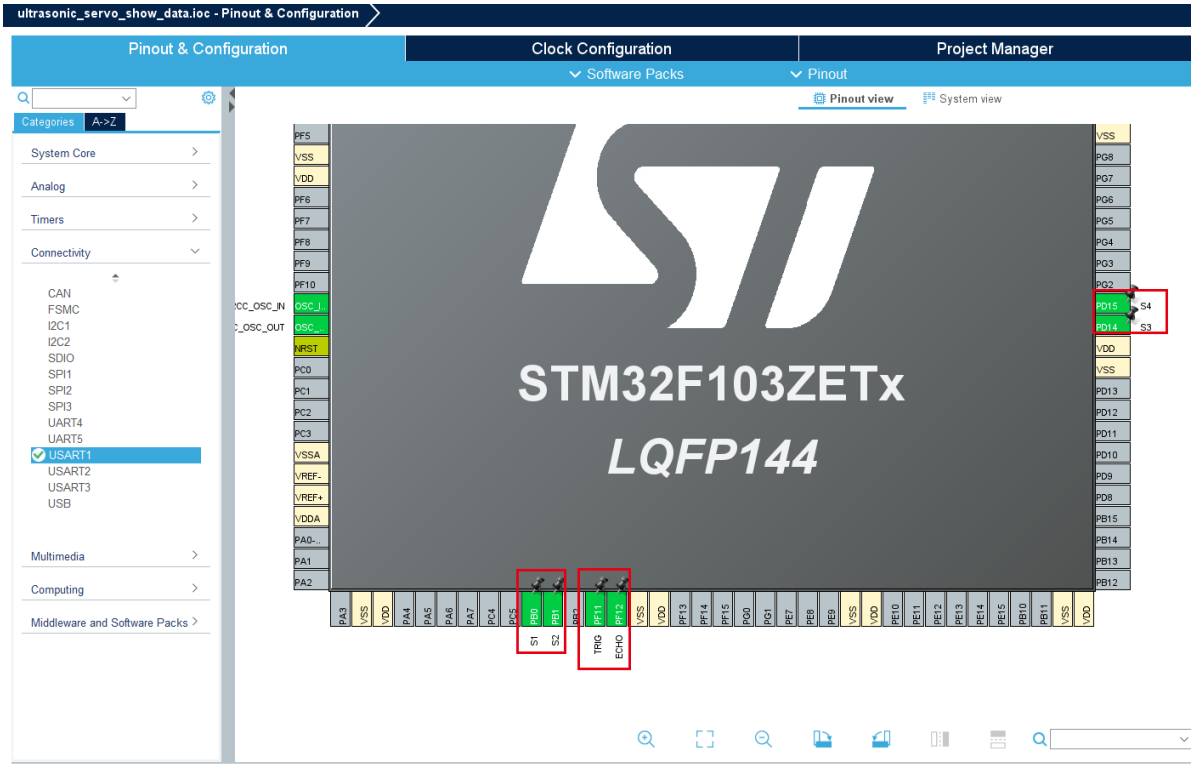
3. Project configuration

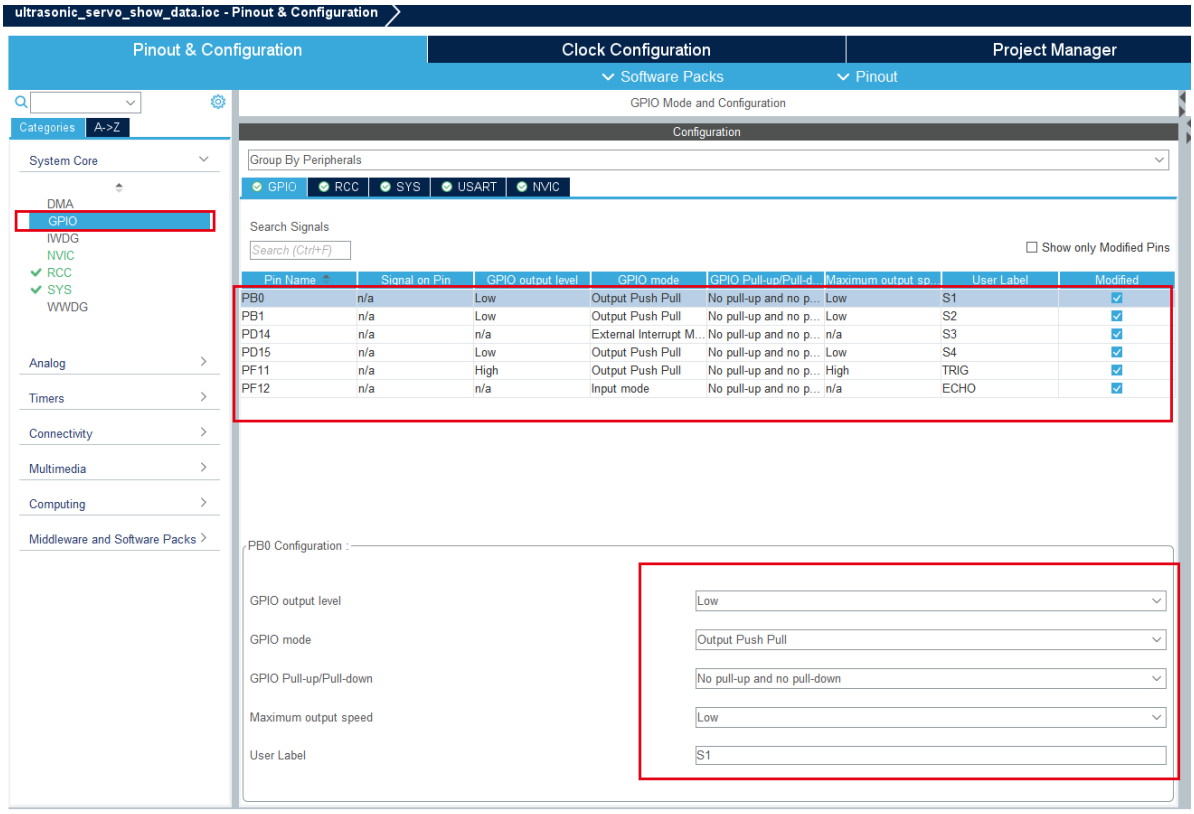
1. Description

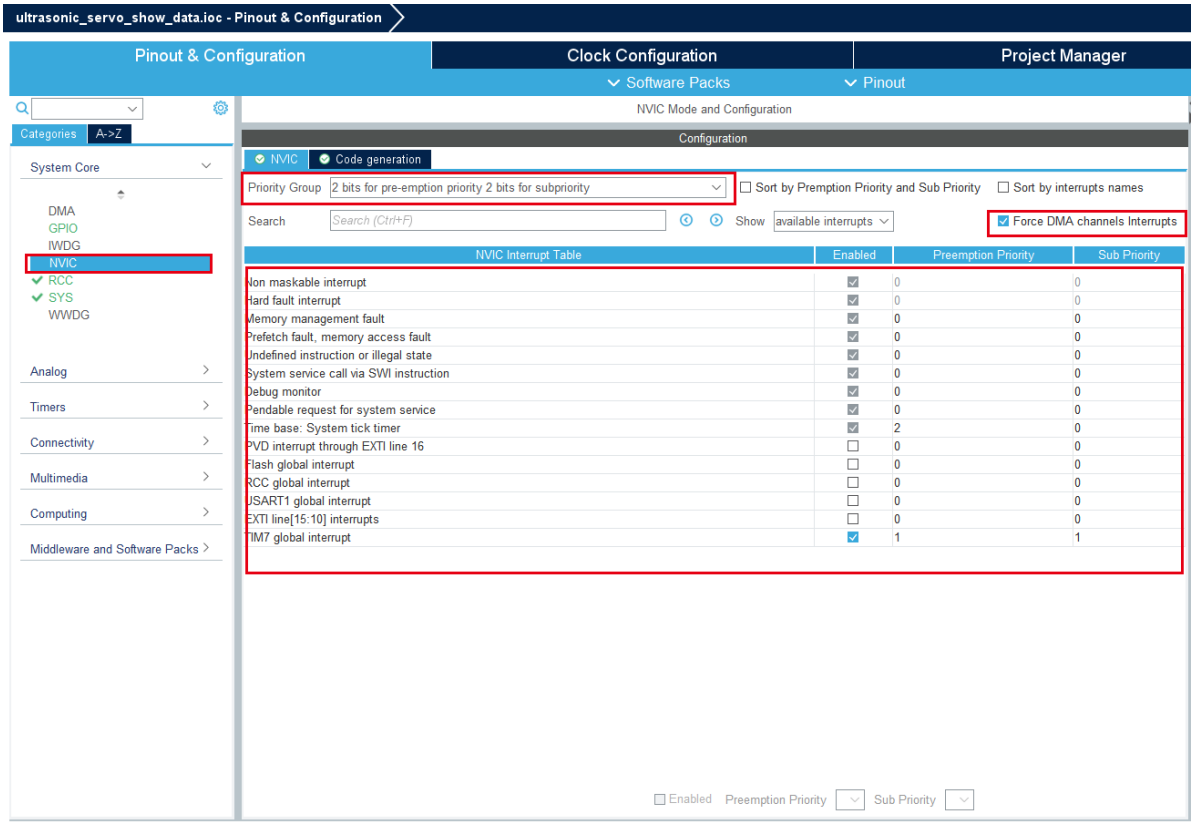
Omitted project configuration part: **New project, chip selection, project configuration, SYS of pin configuration, RCC configuration, clock configuration and project configuration** content

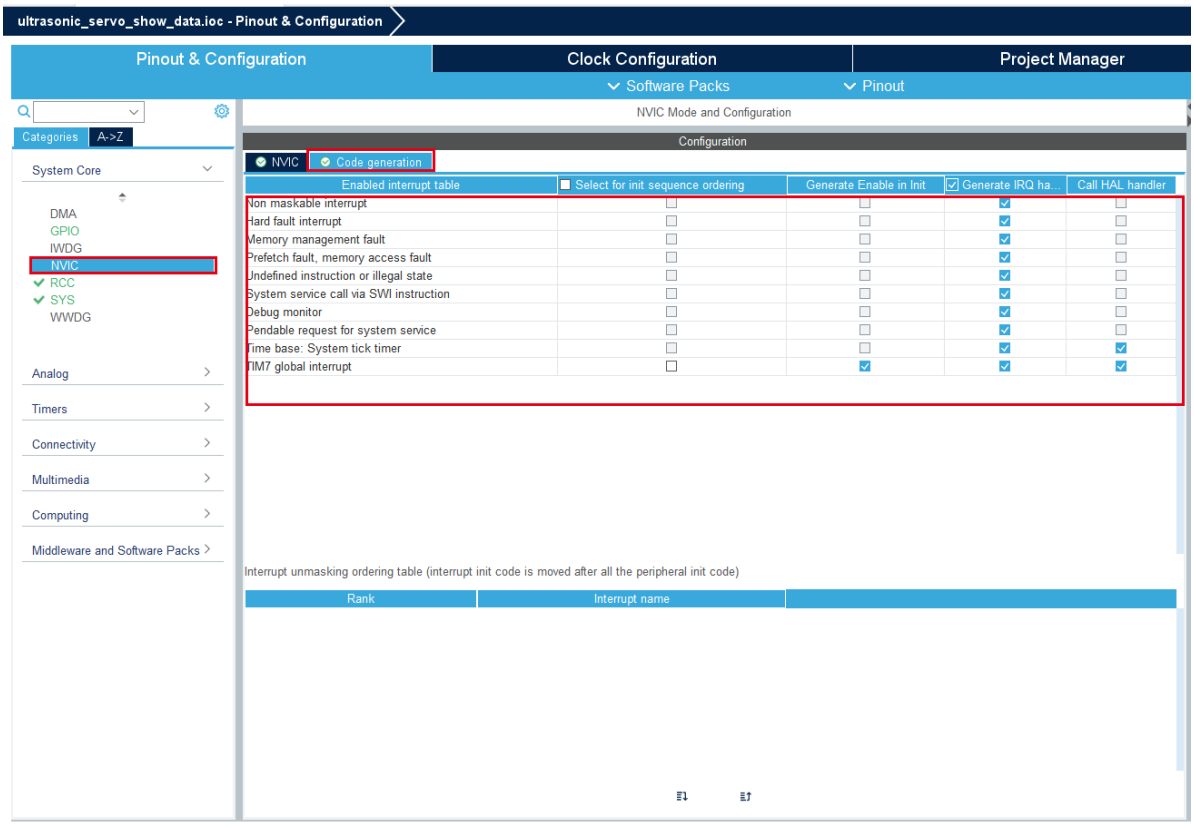
```
Please refer to [2. Development environment construction and use: STM32CubeIDE
installation and use] to understand how to configure the omitted parts of the
project.
```

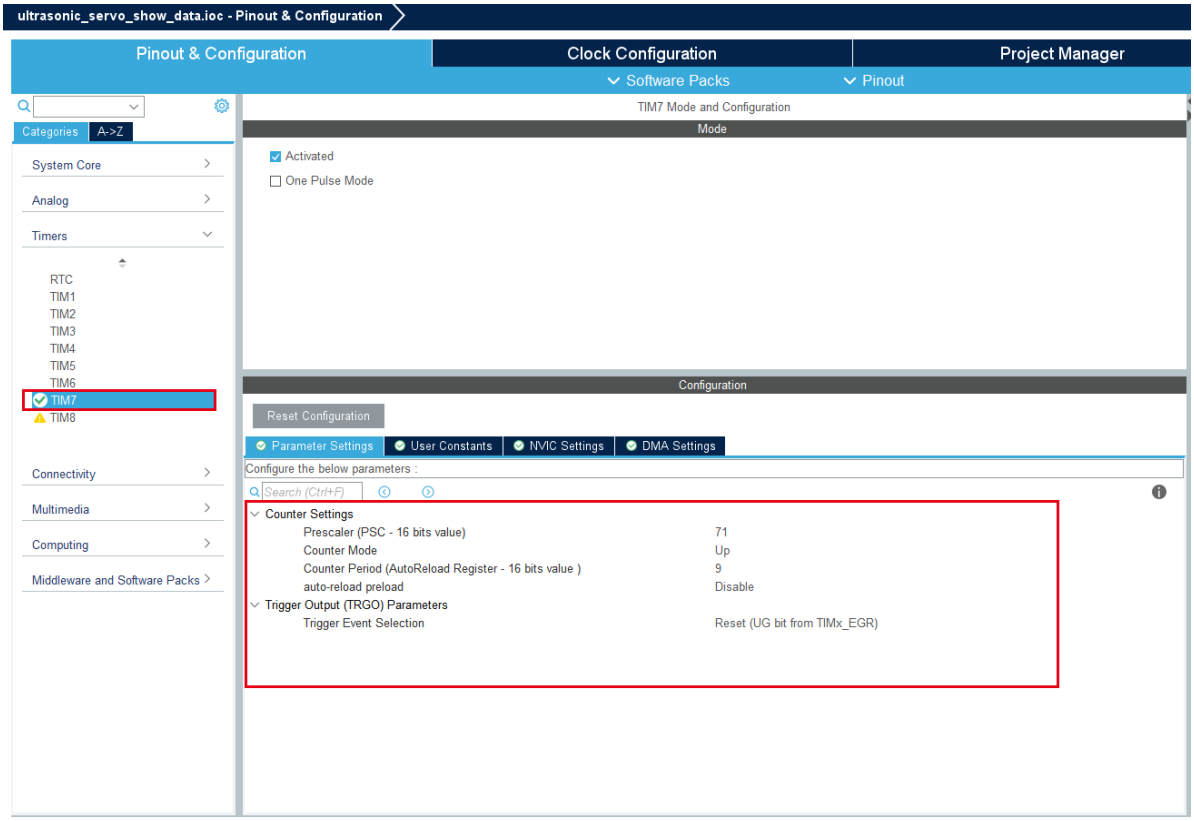
2. Pin configuration

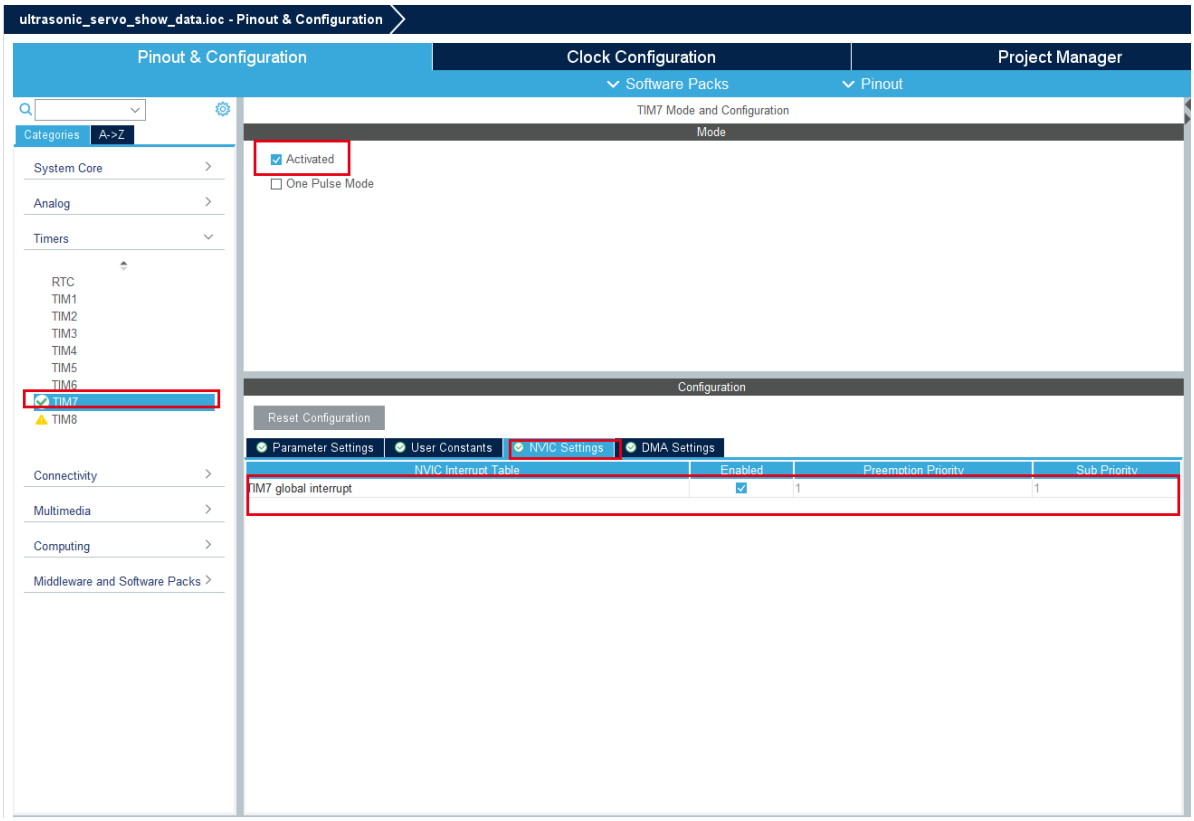

ultrasonic_servo_show_data.ioc - Pinout & Configuration
Pinout & Configuration
Clock Configuration
Project Manager
Software Packs
Pinout
Pinout view
System view
Categories
A->Z
System Core
Analog
Timers
Connectivity
CAN
FSMC
I2C1
I2C2
SDIO
SPI1
SPI2
SPI3
UART4
UART5
USART1
USART2
USART3
USB
Multimedia
Computing
Middleware and Software Packs
STM32F103ZETx
LQFP144
S1
S2
TRIG
ECHO
S3
S4

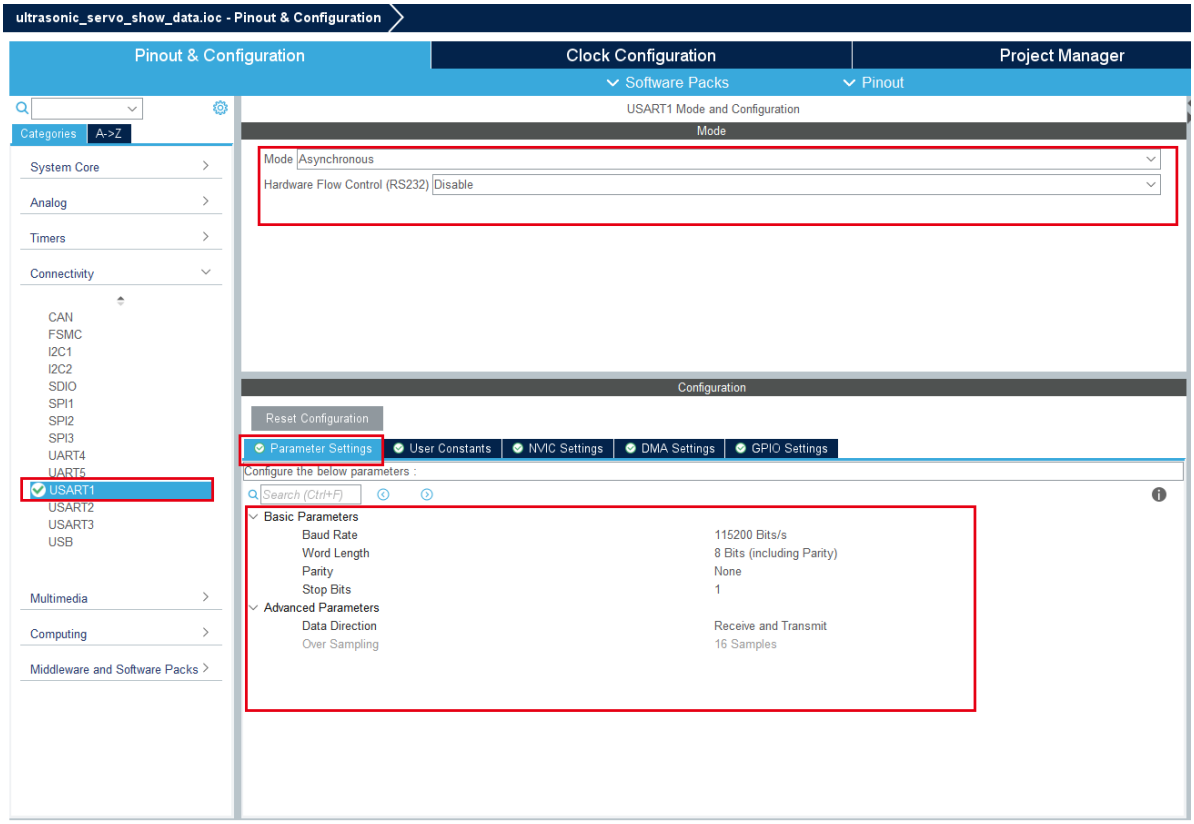

ultrasonic_servo_show_data.ioc - Pinout & Configuration
Pinout & Configuration
Clock Configuration
Project Manager
Software Packs
Pinout
GPIO Mode and Configuration
Configuration
Group By Peripherals
GPIO
RCC
SYS
USART
NVIC
Search Signals
Search (Ctrl+F)
Show only Modified Pins
Pin Name
Signal on Pin
GPIO output level
GPIO mode
GPIO Pull-up/Pull-d...
Maximum output sp...
User Label
Modified
PB0 n/a Low Output Push Pull No pull-up and no p... Low S1
PB1 n/a Low Output Push Pull No pull-up and no p... Low S2
PD14 n/a n/a External Interrupt M... No pull-up and no p... n/a S3
PD15 n/a Low Output Push Pull No pull-up and no p... Low S4
PF11 n/a High Output Push Pull No pull-up and no p... High TRIG
PF12 n/a n/a Input mode No pull-up and no p... n/a ECHO
PB0 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
S1


ultrasonic_servo_show_data.ioc - Pinout & Configuration
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
NVIC Mode and Configuration
Configuration
NVIC
Code generation
Priority Group
2 bits for pre-emption priority 2 bits for subpriority
Sort by Premption Priority and Sub Priority
Sort by interrupts names
Search
Search (Ctrl+F)
Show
available interrupts
Force DMA channels Interrupts
NVIC Interrupt Table
Enabled
Preemption Priority
Sub Priority
Non maskable interrupt
Hard fault interrupt
Memory management fault
Prefetch fault, memory access fault
Undefined instruction or illegal state
System service call via SWI instruction
Debug monitor
Pendable request for system service
Time base: System tick timer
PVD interrupt through EXTI line 16
Flash global interrupt
RCC global interrupt
USART1 global interrupt
EXTI line[15:10] interrupts
TIM7 global interrupt
Enabled
Preemption Priority
Sub Priority


ultrasonic_servo_show_data.ioc - Pinout & Configuration
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
NVIC Mode and Configuration
Configuration
NVIC
Code generation
Enabled interrupt table
Select for init sequence ordering
Generate Enable in Init
Generate IRQ ha...
Call HAL handler
Non maskable interrupt
Hard fault interrupt
Memory management fault
Prefetch fault, memory access fault
Undefined instruction or illegal state
System service call via SWI instruction
Debug monitor
Pendable request for system service
Time base: System tick timer
TIM7 global interrupt
Interrupt unmasking ordering table (interrupt init code is moved after all the peripheral init code)
Rank
Interrupt name


ultrasonic_servo_show_data.ioc - Pinout & Configuration
Pinout & Configuration
Clock Configuration
Project Manager
Software Packs
Pinout
Categories
A->Z
System Core
Analog
Timers
RTC
TIM1
TIM2
TIM3
TIM4
TIM5
TIM6
TIM7
TIM8
Connectivity
Multimedia
Computing
Middleware and Software Packs
TIM7 Mode and Configuration
Mode
Activated
One Pulse Mode
Configuration
Reset Configuration
Parameter Settings
User Constants
NVIC Settings
DMA Settings
Configure the below parameters :
Search (Ctrl+F)
Counter Settings
Prescaler (PSC - 16 bits value) 71
Counter Mode Up
Counter Period (AutoReload Register - 16 bits value) 9
auto-reload preload Disable
Trigger Output (TRGO) Parameters
Trigger Event Selection Reset (UG bit from TIMx_EGR)


ultrasonic_servo_show_data.ioc - Pinout & Configuration
Pinout & Configuration
Clock Configuration
Project Manager
Software Packs
Pinout
Categories
A->Z
System Core
Analog
Timers
RTC
TIM1
TIM2
TIM3
TIM4
TIM5
TIM6
TIM7
TIM8
Connectivity
Multimedia
Computing
Middleware and Software Packs
TIM7 Mode and Configuration
Mode
Activated
One Pulse Mode
Configuration
Reset Configuration
Parameter Settings
User Constants
NVIC Settings
DMA Settings
NVIC Interrupt Table
Enabled
Preemption Priority
Sub Priority
TIM7 global interrupt
1
1



4. Main functions

According to our tutorial STM32CubeIDE can generate the corresponding gpio.c, gpio.h, tim.c and tim.h files. For later transplantation and peripheral module driver, we will place the automatically generated code in the BSP under the project file. folder.

1. User function

> **Function: Get_distance**

Function prototype	**float Get_distance(void)**
Function description	**Get the average of 5 ultrasonic measurement data**
Input parameters	**None**
Return value	**Processed distance**

> **Function: PwmServo_Set_Angle**

Function prototype	**void PwmServo_Set_Angle(uint8_t index, uint8_t angle)**
Function description	**Ultrasonic range detection**
Input parameter 1	**Servo serial number: 0~MAX_PWM_SERVO-1**
Input parameter 2	**Angle value: 0-180**
Return value	**None**

> **Function: Bsp_TIM7_Init**

Function prototype	void Bsp_TIM7_Init(void)
Function description	**Open timer terminal**
Input parameters	**None**
Return value	**None**

> **Function: Bsp_UART1_Init**

Function prototype	void Bsp_UART1_Init(void)
Function description	**Initialize serial port 1**
Input parameters	**None**
Return value	**None**

5. Experimental phenomena

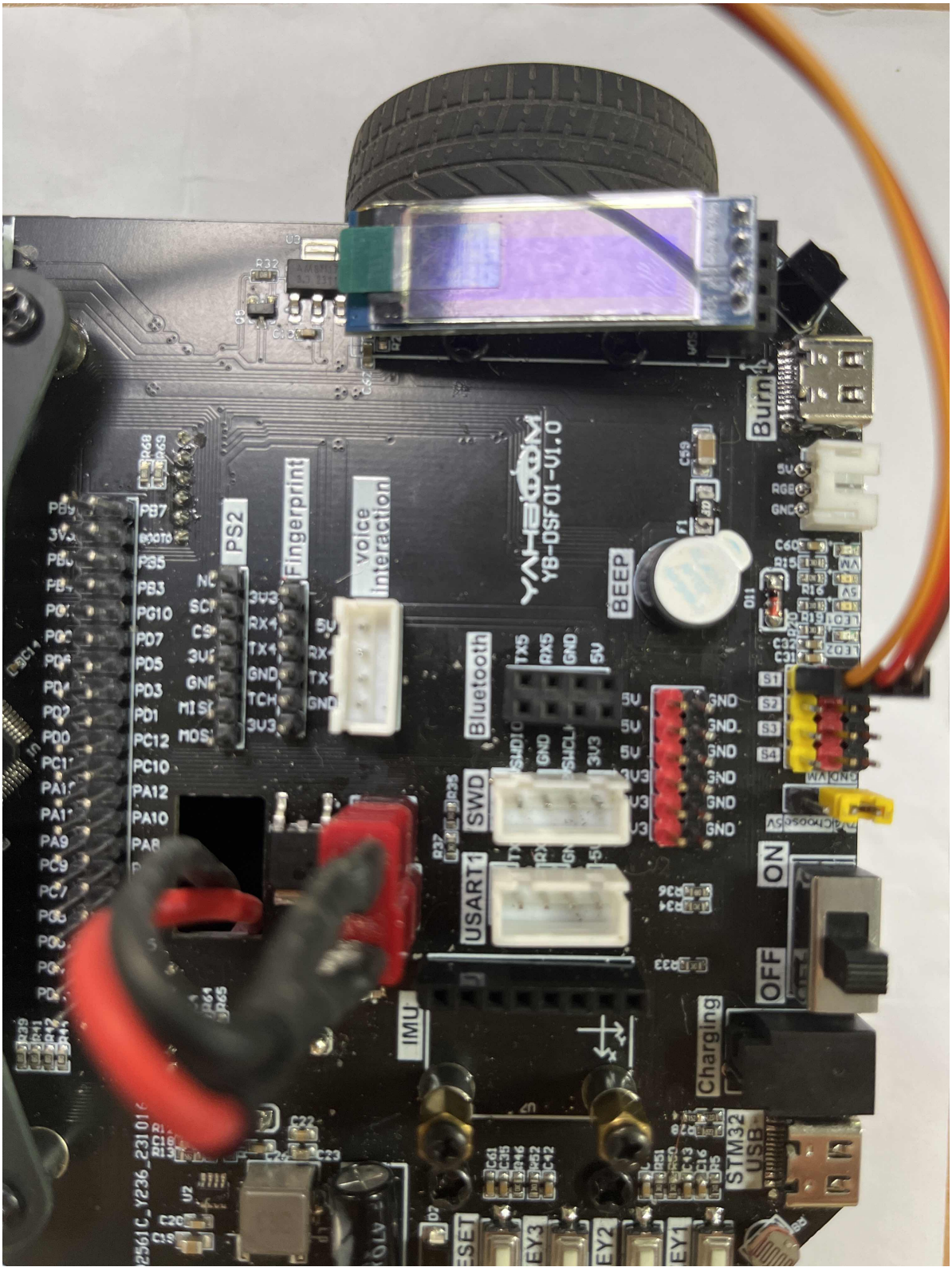
During installation, you need to center the servo before installing it; the [ultrasonic_servo_calibrating.hex] file is provided in the tutorial folder for calibration. After burning this program, the servo will rotate to 90°. At this time, the ultrasonic and servo clouds are installed. Connect the connecting plate of the platform (the actual picture is shown below) to the servo. When installing the ultrasonic wave, it needs to face the front (it will have no effect if it is slightly off).

After the servo is calibrated, we can burn the code in this chapter. Then after powering up the car, press the Reset button. We need to connect to the host computer through the type C interface, open the serial port assistant, and set the parameters as shown in the figure below. Then we can view the measured ultrasonic distance through the serial port assistant.

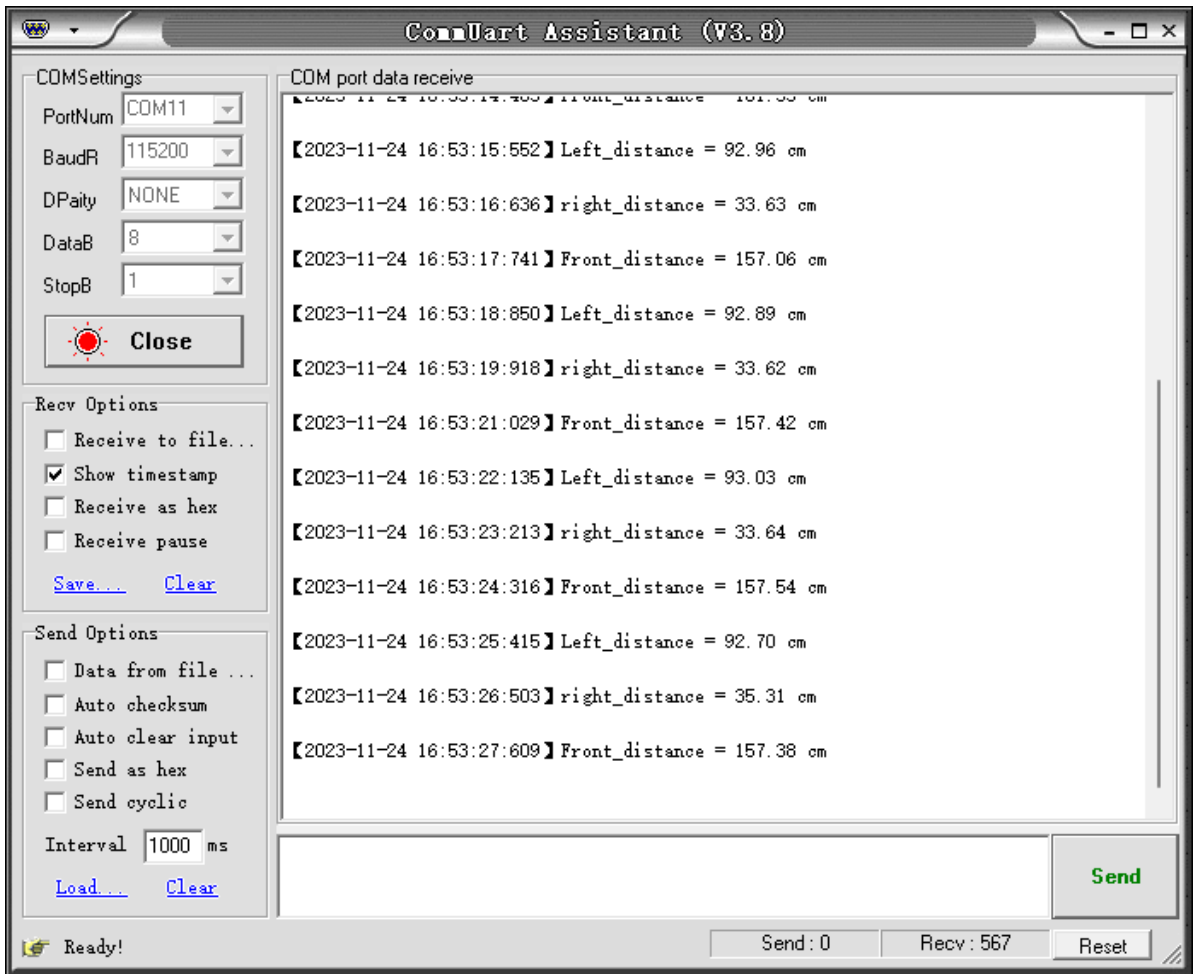
After the car is powered on, the gimbal will first turn to the middle, then to the left, and then to the right. Cycle back and forth in these three directions. The serial port will print the distance between the ultrasonic measurement and the obstacle.

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

The servo wiring is as follows:



The effect is as follows:


ComUart Assistant (V3.8)
COMSettings
PortNum COM11
BaudR 115200
DPaity NONE
DataB 8
StopB 1
Close
Recv Options
Receive to file...
Show timestamp
Receive as hex
Receive pause
Save...
Clear
Send Options
Data from file ...
Auto checksum
Auto clear input
Send as hex
Send cyclic
Interval 1000 ms
Load...
Clear
COM port data receive
【2023-11-24 16:53:15:552】Left_distance = 92.96 cm
【2023-11-24 16:53:16:636】right_distance = 33.63 cm
【2023-11-24 16:53:17:741】Front_distance = 157.06 cm
【2023-11-24 16:53:18:850】Left_distance = 92.89 cm
【2023-11-24 16:53:19:918】right_distance = 33.62 cm
【2023-11-24 16:53:21:029】Front_distance = 157.42 cm
【2023-11-24 16:53:22:135】Left_distance = 93.03 cm
【2023-11-24 16:53:23:213】right_distance = 33.64 cm
【2023-11-24 16:53:24:316】Front_distance = 157.54 cm
【2023-11-24 16:53:25:415】Left_distance = 92.70 cm
【2023-11-24 16:53:26:503】right_distance = 35.31 cm
【2023-11-24 16:53:27:609】Front_distance = 157.38 cm
Send
Ready!
Send : 0
Recv : 567
Reset