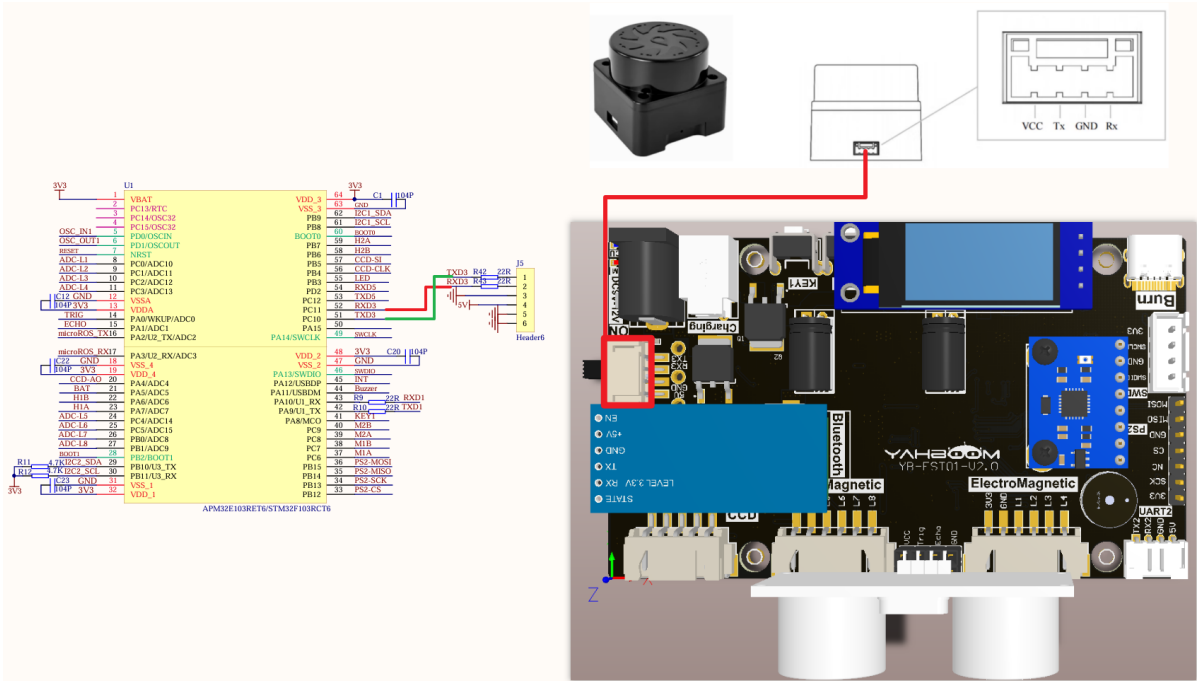
Tmini-Plus lidar-Reading data (USART)

The tutorial demonstrates how to receive radar data via serial port 3 and print the distance values corresponding to each angle of the radar via serial port 1.

```
The tutorial only introduces the standard library project code
```

Hardware connection

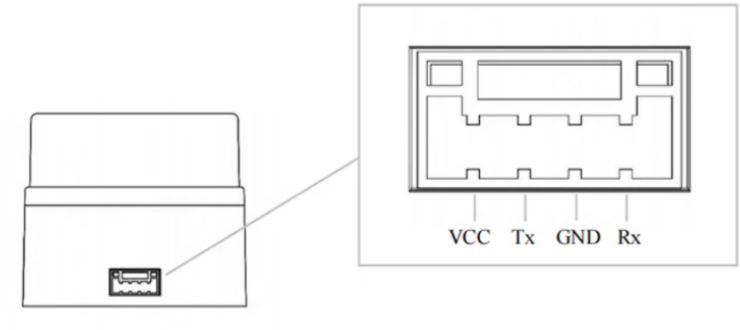


Since we have configured a special connection line, we only need to install it to the corresponding interface:

Peripherals	Development board
Tmini-Plus radar: VCC	5V
Tmini-Plus radar: TXD	PC10
Tmini-Plus radar: RXD	PC11
Tmini-Plus radar: GND	GND

Control principle

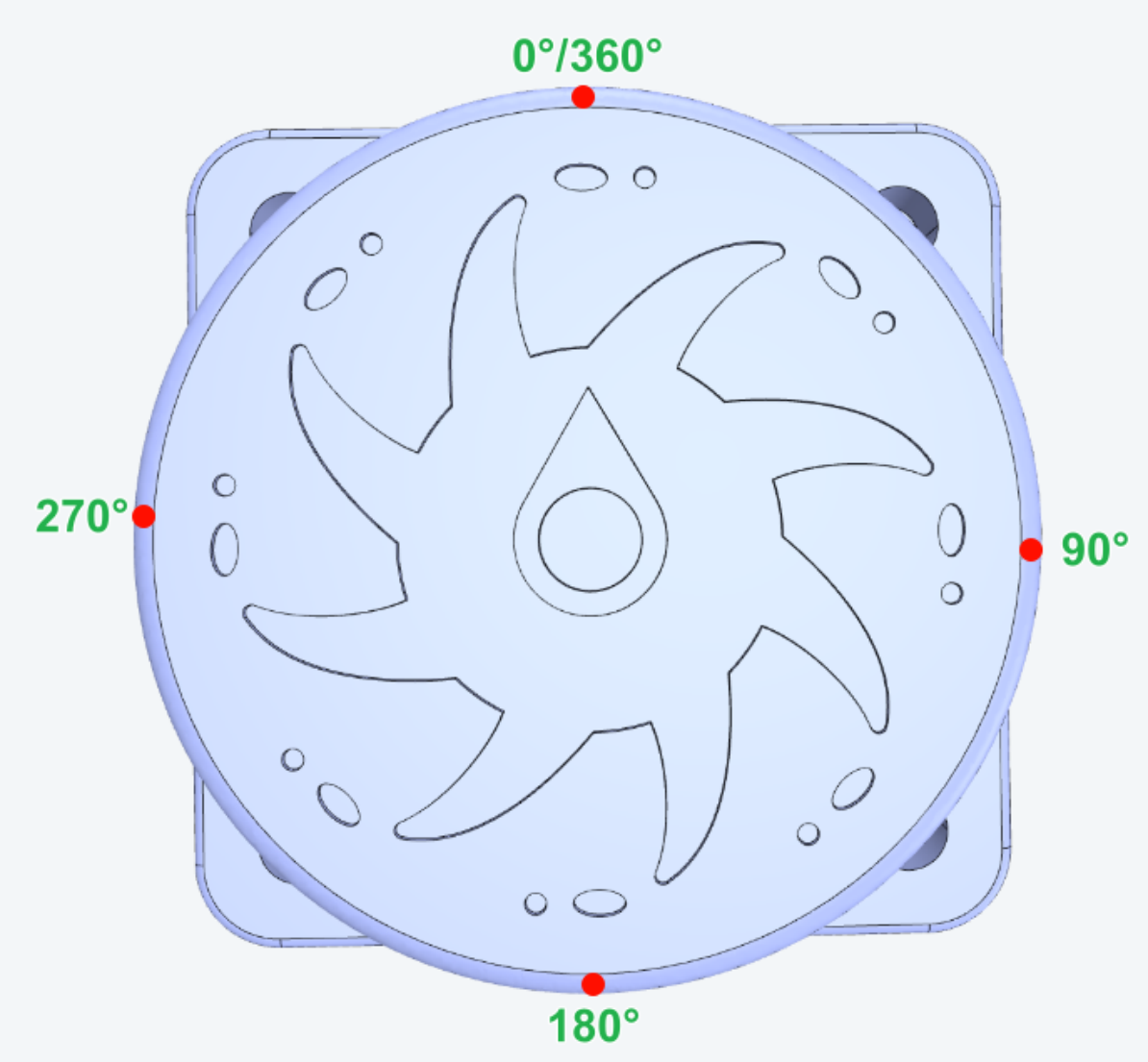
- Tmini-Plus radar



Product name	**Tmini-Plus radar**
Scanning frequency	6-12Hz
Sampling frequency	4000 times/s
Measuring radius	Black object: 12m
Minimum measuring distance	0.05m
Ranging principle	TOF ranging
Scanning angle	360°
Communication interface	Standard asynchronous serial port (UART) 1. Baud rate: 230400 2. Data bits: 8 3. Check bit: None 4. Stop bit: 1
ROS support	ROS1/ROS2
Windows support	Host computer

> ** Radar angle distribution**

The arrow in the center of the radar points to 0°/360°, and the angle increases clockwise.



> Communication protocol

```
For detailed information, please refer to the "T_Mini_Plus Manual"
```

Software configuration

Pin definition

Main control chip	Pin	Main function (after reset)	Default multiplexing function	Redefine function
STM32F103RCT6	PC10	PC10	USART4_TX/SDIO_D2	USART3_TX
STM32F103RCT6	PC11	PC11	USART4_RX/SDIO_D3	USART3_RX

Software code

Since the default pin function is a normal IO pin function, we need to use the redefine function.

```
Product supporting materials source code path: Attachment → Source code summary → 2. Extended_Course → 12. Tmini-Plus_Ladar
```

Control function

The tutorial only briefly introduces the code, you can open the project source code to read it in detail.

> **USART3_init**

```
void USART3_init(u32 baudrate)
{
    GPIO_InitTypeDef GPIO_InitStructure;
    USART_InitTypeDef USART_InitStructure;
    NVIC_InitTypeDef NVIC_InitStructure;

    RCC_APB1PeriphClockCmd(RCC_APB1Periph_USART3, ENABLE);
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOC |RCC_APB2Periph_AFIO, ENABLE);

    GPIO_PinRemapConfig(GPIO_PartialRemap_USART3, ENABLE);

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_Init(GPIOC, &GPIO_InitStructure);

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_11;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;
    GPIO_Init(GPIOC, &GPIO_InitStructure);

    USART_InitStructure.USART_BaudRate = baudrate;
    USART_InitStructure.USART_WordLength = USART_WordLength_8b;
    USART_InitStructure.USART_StopBits = USART_StopBits_1;
    USART_InitStructure.USART_Parity = USART_Parity_No ;
    USART_InitStructure.USART_HardwareFlowControl =
USART_HardwareFlowControl_None;
    USART_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
    USART_Init(USART3, &USART_InitStructure);
    USART_ITConfig(USART3, USART_IT_TXE, DISABLE);
    USART_ITConfig(USART3, USART_IT_RXNE, ENABLE);
    //USART_ClearFlag(USART3,USART_FLAG_TC);
    USART_Cmd(USART3, ENABLE);

    NVIC_InitStructure.NVIC_IRQChannel = USART3_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 3;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 1;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);
}
```

> **USART3_IRQHandler**

```
void USART3_IRQHandler(void)
{
    uint8_t Rx3_Temp;
    if (USART_GetITStatus(USART3, USART_IT_RXNE) != RESET)
    {
        USART_ClearITPendingBit(USART3, USART_IT_RXNE);
        Rx3_Temp = USART_ReceiveData(USART3);
        recv_lidar_data(Rx3_Temp);
    }
}
```

> **recv_lidar_data**

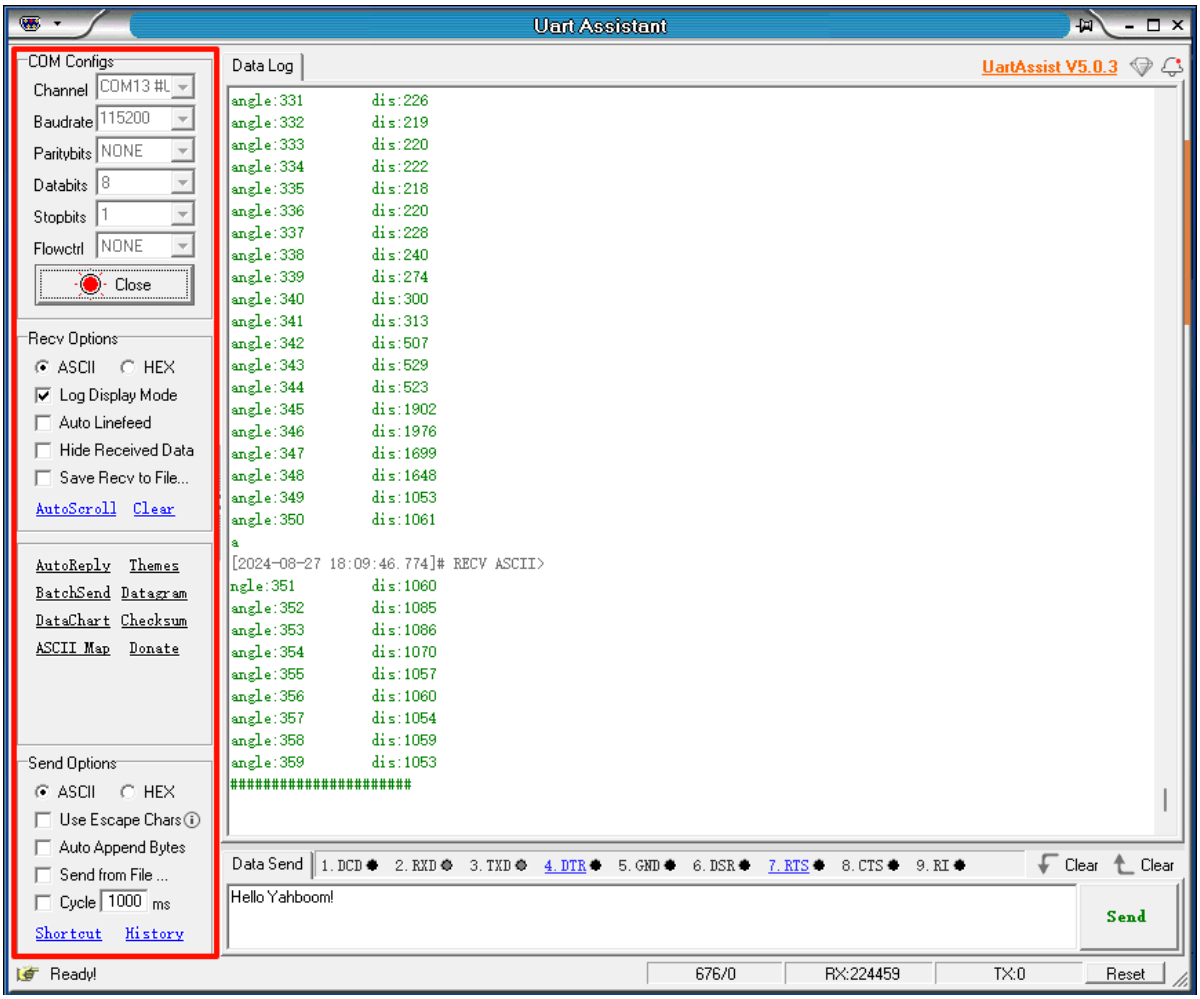
```
void recv_lidar_data(u8 rxtemp)
{
    static uint8_t step = 0;
    static uint16_t si_len = 0;
    static uint8_t si_index = 0;
    switch(step)
    {

        case 0:
            if(rxtemp == Lidar_HeaderLSB)
            {
                step = 1;
                g_recvbuf[0] = Lidar_HeaderLSB;
            }
        break;

        case 1:
            if(rxtemp == Lidar_HeaderMSB)
            {
                step = 2;
                g_recvbuf[1] = Lidar_HeaderMSB;
            }
        break;

        case 2: g_recvbuf[step] = rxtemp; step++; break; // CT Information
        case 3: g_recvbuf[step] = rxtemp; step++; si_len = rxtemp * 3; // Number
of S

            // Here we judge whether the data length is greater than the cache.
            if(si_len+10>=TempLen_Max)
            {
                    si_len = 0;
                    step = 0;
                    memset(g_recvbuf,0,sizeof(g_recvbuf));
            }
            break;

        case 4: g_recvbuf[step] = rxtemp; step++;  break; //Start angle low 8
bits
        case 5: g_recvbuf[step] = rxtemp; step++;  break; //Start angle high 8
bits
```

```
        case 6: g_recvbuf[step] = rxtemp; step++;  break; //End angle low 8 bits
        case 7: g_recvbuf[step] = rxtemp; step++;  break; //End angle high 8 bits
        case 8: g_recvbuf[step] = rxtemp; step++;  break; //Check code low 8 bits
        case 9: g_recvbuf[step] = rxtemp; step++;  break; //Check code high 8
bits

        case 10:
        {
                g_recvbuf[step + si_index] = rxtemp;
                si_index++;

                if(si_index >= si_len )
                {
                    Deal_Radar();
                    si_index = 0;
                    si_len = 0;
                    step = 0;
                    memset(g_recvbuf,0,sizeof(g_recvbuf));
                }
                break;
        }
    }
}
```

Experimental phenomenon

The Tmini-Plus_Ladar.hex file generated by the project compilation is located in the OBJ folder of the Tmini-Plus_Ladar project. Find the Tmini-Plus_Ladar.hex file corresponding to the project and use the FlyMcu software to download the program into the development board.

After the program is successfully downloaded: the serial port will print the radar angle and its corresponding distance.


Uart Assistant
COM Configs
Channel COM13 #L
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
angle:331 dis:226
angle:332 dis:219
angle:333 dis:220
angle:334 dis:222
angle:335 dis:218
angle:336 dis:220
angle:337 dis:228
angle:338 dis:240
angle:339 dis:274
angle:340 dis:300
angle:341 dis:313
angle:342 dis:507
angle:343 dis:529
angle:344 dis:523
angle:345 dis:1902
angle:346 dis:1976
angle:347 dis:1699
angle:348 dis:1648
angle:349 dis:1053
angle:350 dis:1061
a
[2024-08-27 18:09:46.774]# RECV ASCII>
ngle:351 dis:1060
angle:352 dis:1085
angle:353 dis:1086
angle:354 dis:1070
angle:355 dis:1057
angle:356 dis:1060
angle:357 dis:1054
angle:358 dis:1059
angle:359 dis:1053
######################
Data Send
1. DCD 2. RXD 3. TXD 4. DTR 5. GND 6. DSR 7. RTS 8. CTS 9. RI
Clear Clear
Hello Yahboom!
Send
Ready!
676/0 RX:224459 TX:0 Reset