Ultrasonic follow

The tutorial mainly demonstrates the following function of the balance car combined with the ultrasonic module.

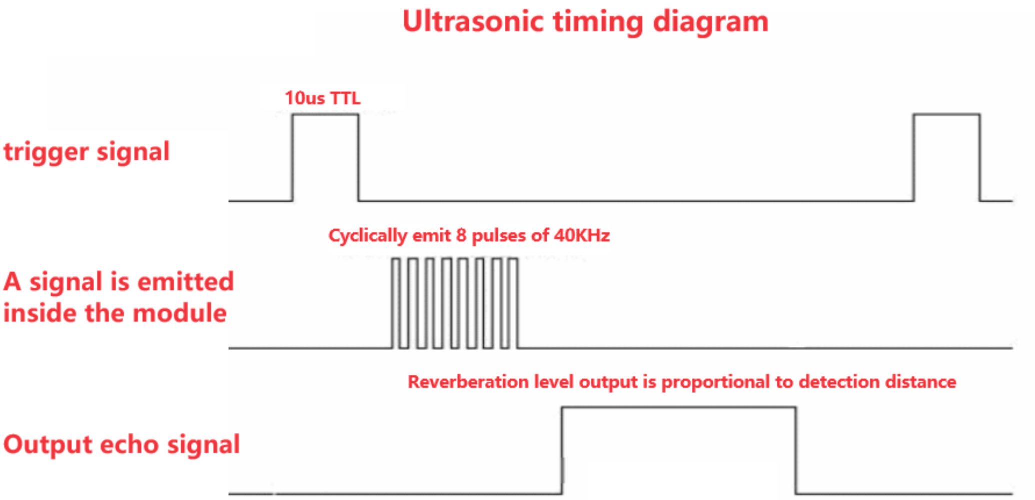
```
The tutorial only introduces the standard library project code
```

Hardware connection

Ultrasonic module	STM32F103RCT6
VCC	5V
TRIG	PA0
ECHO	PA1
GND	GND

Distance measurement principle

The ultrasonic module is mainly responsible for obtaining the distance between the obstacle and the car to realize the following function.


Ultrasonic timing diagram
10us TTL
trigger signal
Cyclically emit 8 pulses of 40KHz
A signal is emitted inside the module
Reverberation level output is proportional to detection distance
Output echo signal

> Trigger ranging signal

The TRIG pin sends a high level signal for more than 10us.

```
The ultrasonic module will automatically send 8 40KHz square waves and
automatically detect whether there is a signal return
```

> **Receive echo signal**

If there is a signal return, the ECHO pin will output a high level. The duration of the high level is the time from the ultrasonic wave transmission to the return.

> **Distance conversion**

$$D_{istance} = \frac{T_{high} * V_{speed}}{2}$$

```
T: high level time of echo signal
V: sound speed (approximately equal to 340m/s)
```

Main code

The tutorial mainly explains the code for the following function. For detailed code, please refer to the corresponding project file.

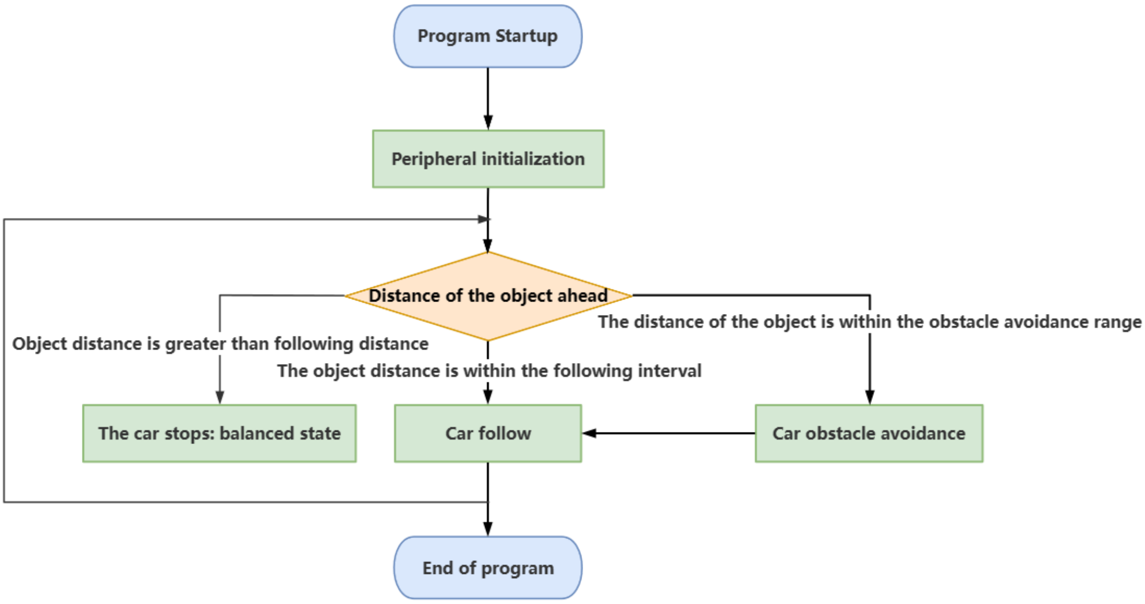
> **App_Change_Car**

Control the car according to the ultrasonic distance to achieve obstacle avoidance and following actions.

```
void App_Change_Car(void)
{
    //The car is not in a stationary state or ultrasonic state
    if(g_newcarstate != enSTOP && g_newcarstate != enAvoid && g_newcarstate != enFollow)
    {
        return;
    }
    //If the ultrasonic is not plugged in or abnormal
    if(g_distance == 0)
    {
        return;
    }
    if(g_distance >20 && g_distance<100)
    {
        g_newcarstate = enAvoid;//Obstacle avoidance
    }
    else if(g_distance >240 && g_distance<300)
    {
        g_newcarstate = enFollow;//Follow
    }
    else
    {
        g_newcarstate = enSTOP;//Stop state
    }
}
```

Program flow chart

Briefly introduce the process of function implementation:


Program Startup
Peripheral initialization
Distance of the object ahead
Object distance is greater than following distance
The object distance is within the following interval
The distance of the object is within the obstacle avoidance range
The car stops: balanced state
Car follow
Car obstacle avoidance
End of program

Experimental phenomenon

> ### Software code

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

```
Product supporting materials source code path: Attachment → Source code summary → 4.Balanced_Car_base → 03.U_Follow
```

> ### Experimental phenomenon

After the program is started, press KEY1 according to the OLED prompt to start the balance car follow function: OLED will display the distance of the object in front and the inclination of the balance car in real time!

```
The program has voltage detection. If the voltage is less than 9.6V, the low voltage alarm will be triggered and the buzzer will sound.
Common situations for triggering voltage alarms:
1. The power switch of the development board is not turned on, and only the Type-C data cable is connected for power supply
2. The battery pack voltage is lower than 9.6V and needs to be charged in time
```