6.5 Buzzer play music

1. Learning goal:

Learn about the how to make passive buzzer play music.

2. Experimental phenomena:

After power is turned on, buzzer will sound. Then, when we press K1 button for the first time, it will play Star; when we press K1 button for the second time, it will play Bingo; when we press K1 button for the third time, it will play Merry Christmas; when we press K1 button for the fourth time, it will play Birthday; when we press K1 button for the fifth time, stop play.

3. About buzzer

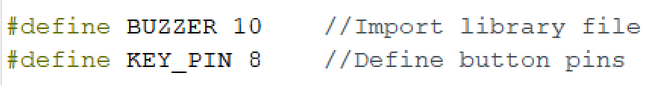
There are two types of buzzer, one is an active buzzer and the other is a passive buzzer. The active buzzer is relatively simple, and can be sounded directly with a high level, but with only one tone, the frequency and rhythm cannot be adjusted.

Passive buzzer directly for high level is not working properly, you need to give a certain frequency of the signal, the frequency and rhythm can be adjusted according to the code.

The Omniduino robot car uses a passive buzzer.

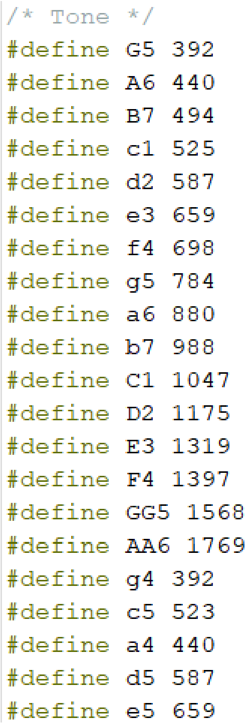
4. Code analysis

4.1 Define buzzer and K1 button pins

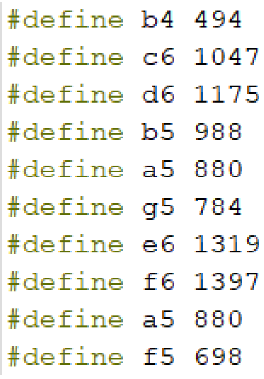


```
#define BUZZER 10     //Import library file
#define KEY_PIN 8     //Define button pins
```

4.2 Define tone:

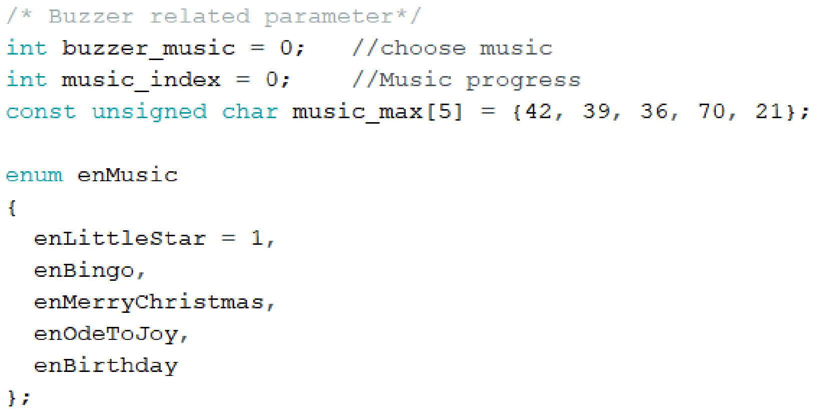


```
/* Tone */
#define G5 392
#define A6 440
#define B7 494
#define c1 525
#define d2 587
#define e3 659
#define f4 698
#define g5 784
#define a6 880
#define b7 988
#define C1 1047
#define D2 1175
#define E3 1319
#define F4 1397
#define GG5 1568
#define AA6 1769
#define g4 392
#define c5 523
#define a4 440
#define d5 587
#define e5 659
```



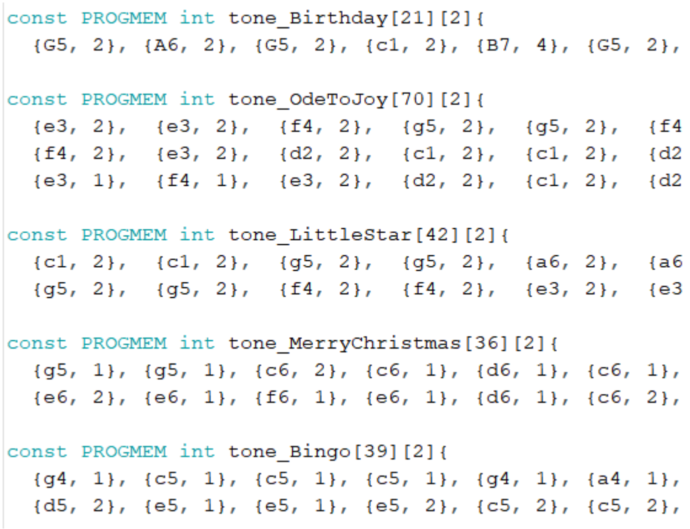
```
#define b4 494
#define c6 1047
#define d6 1175
#define b5 988
#define a5 880
#define g5 784
#define e6 1319
#define f6 1397
#define a5 880
#define f5 698
```

4.3 Define buzzer parameter:



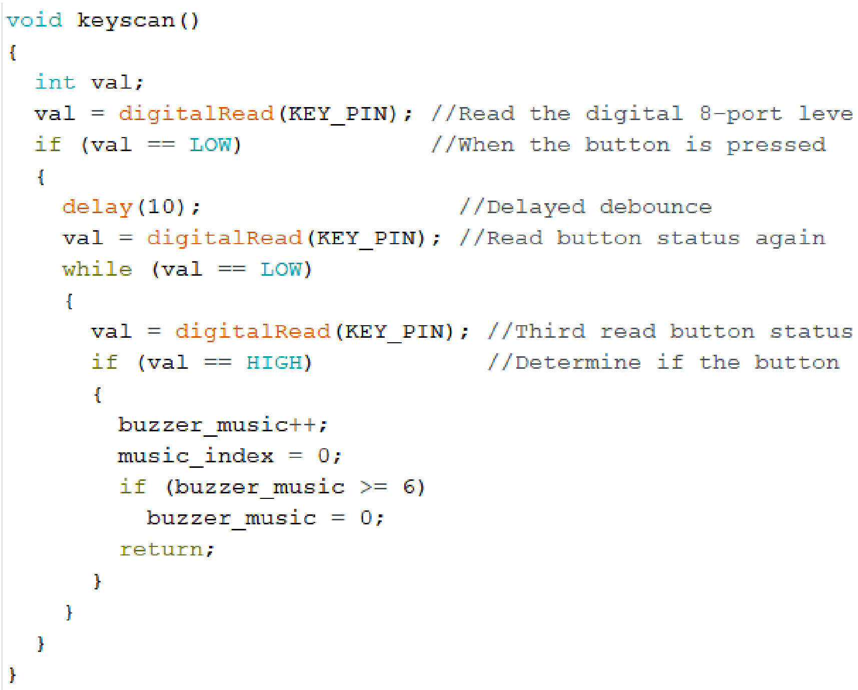
```
/* Buzzer related parameter*/
int buzzer_music = 0;   //choose music
int music_index = 0;    //Music progress
const unsigned char music_max[5] = {42, 39, 36, 70, 21};

enum enMusic
{
  enLittleStar = 1,
  enBingo,
  enMerryChristmas,
  enOdeToJoy,
  enBirthday
};
```

4.4 Define a notation for each piece of music, and then play it in this order to become a song.



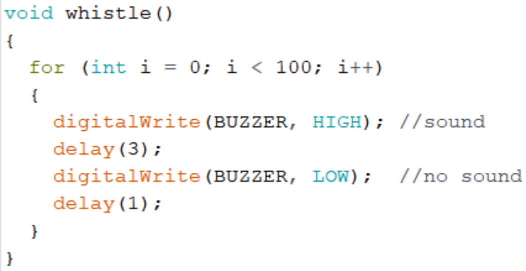
```
const PROGMEM int tone_Birthday[21][2]{
  {G5, 2}, {A6, 2}, {G5, 2}, {c1, 2}, {B7, 4}, {G5, 2},

const PROGMEM int tone_OdeToJoy[70][2]{
  {e3, 2},  {e3, 2},  {f4, 2},  {g5, 2},  {g5, 2},  {f4
  {f4, 2},  {e3, 2},  {d2, 2},  {c1, 2},  {c1, 2},  {d2
  {e3, 1},  {f4, 1},  {e3, 2},  {d2, 2},  {c1, 2},  {d2

const PROGMEM int tone_LittleStar[42][2]{
  {c1, 2},  {c1, 2},  {g5, 2},  {g5, 2},  {a6, 2},  {a6
  {g5, 2},  {g5, 2},  {f4, 2},  {f4, 2},  {e3, 2},  {e3

const PROGMEM int tone_MerryChristmas[36][2]{
  {g5, 1}, {g5, 1}, {c6, 2}, {c6, 1}, {d6, 1}, {c6, 1},
  {e6, 2}, {e6, 1}, {f6, 1}, {e6, 1}, {d6, 1}, {c6, 2},

const PROGMEM int tone_Bingo[39][2]{
  {g4, 1}, {c5, 1}, {c5, 1}, {c5, 1}, {g4, 1}, {a4, 1},
  {d5, 2}, {e5, 1}, {e5, 1}, {e5, 2}, {c5, 2}, {c5, 2},
```

4.5 New create keyscan() function, each time press the function button buzzer_music add 1, control to switch the music.



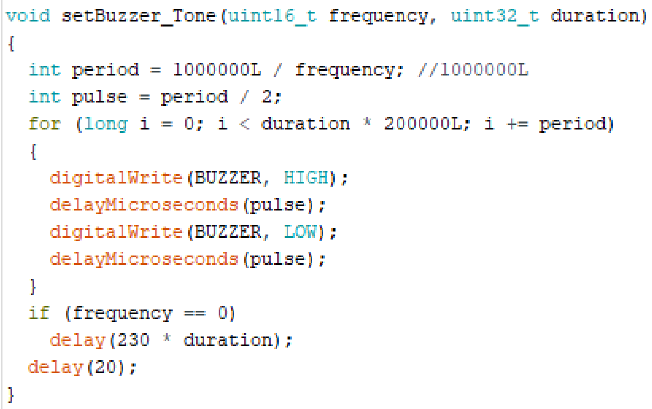
```
void keyscan()
{
  int val;
  val = digitalRead(KEY_PIN); //Read the digital 8-port leve
  if (val == LOW)             //When the button is pressed
  {
    delay(10);                  //Delayed debounce
    val = digitalRead(KEY_PIN); //Read button status again
    while (val == LOW)
    {
      val = digitalRead(KEY_PIN); //Third read button status
      if (val == HIGH)            //Determine if the button
      {
        buzzer_music++;
        music_index = 0;
        if (buzzer_music >= 6)
          buzzer_music = 0;
        return;
      }
    }
  }
}
```

4.6 New create whistle() function, play whistle.



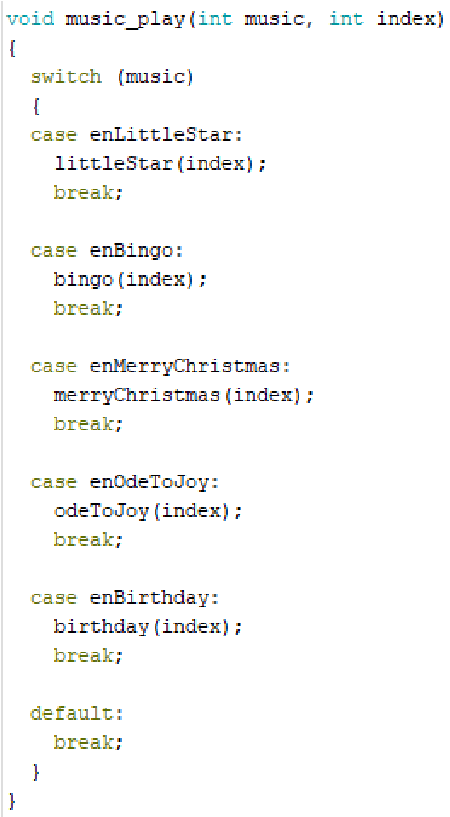
```
void whistle()
{
  for (int i = 0; i < 100; i++)
  {
    digitalWrite(BUZZER, HIGH); //sound
    delay(3);
    digitalWrite(BUZZER, LOW);  //no sound
    delay(1);
  }
}
```

4.7 New create setBuzzer_Tone() function, be used to play tone, first parameter frequency, second parameter is duration.



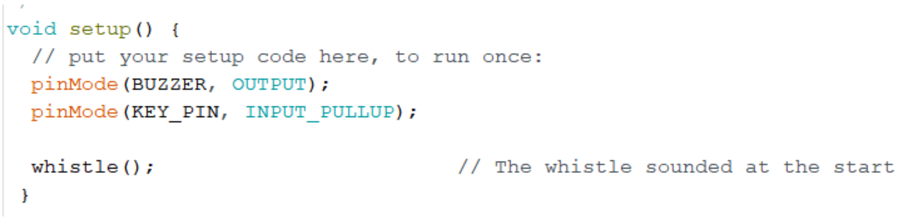
```
void setBuzzer_Tone(uint16_t frequency, uint32_t duration)
{
  int period = 1000000L / frequency; //1000000L
  int pulse = period / 2;
  for (long i = 0; i < duration * 200000L; i += period)
  {
    digitalWrite(BUZZER, HIGH);
    delayMicroseconds(pulse);
    digitalWrite(BUZZER, LOW);
    delayMicroseconds(pulse);
  }
  if (frequency == 0)
    delay(230 * duration);
  delay(20);
}
```

4.8 New create music_play() function, paly music.



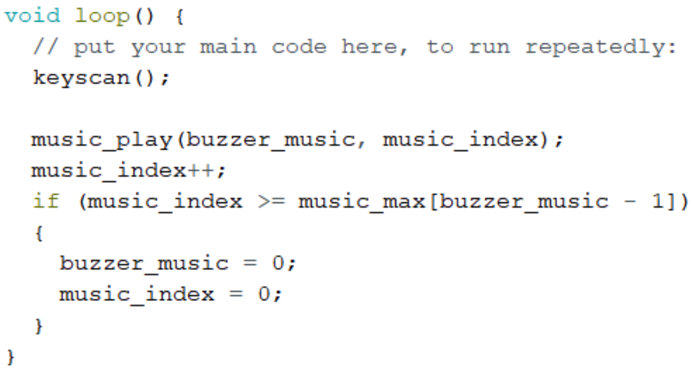
```
void music_play(int music, int index)
{
  switch (music)
  {
  case enLittleStar:
    littleStar(index);
    break;

  case enBingo:
    bingo(index);
    break;

  case enMerryChristmas:
    merryChristmas(index);
    break;

  case enOdeToJoy:
    odeToJoy(index);
    break;

  case enBirthday:
    birthday(index);
    break;

  default:
    break;
  }
}
```

4.9 Set the pin mode in the setup() function and let the buzzer sound



```
void setup() {
  // put your setup code here, to run once:
  pinMode(BUZZER, OUTPUT);
  pinMode(KEY_PIN, INPUT_PULLUP);

  whistle();                        // The whistle sounded at the start
 }
```

4.10 In the loop() function detect button, each time you press to switch songs, and play the song from the beginning.



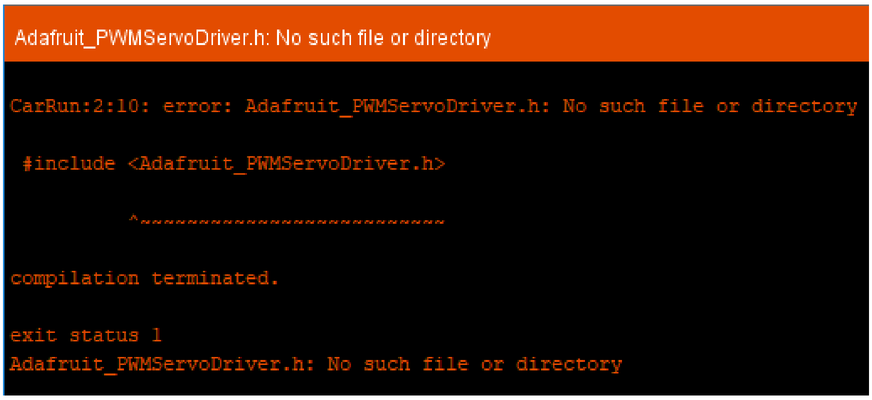
```
void loop() {
  // put your main code here, to run repeatedly:
  keyscan();

  music_play(buzzer_music, music_index);
  music_index++;
  if (music_index >= music_max[buzzer_music - 1])
  {
    buzzer_music = 0;
    music_index = 0;
  }
}
```

5. Compiling and downloading code

5.1 Open the **Buzzer.ino** program, select the serial port and click upload directly (the Omniduino car must first be connected to the computer via the USB data cable).

5.2 If there is an error like the following, it means that the library file is missing. Please copy the library file provided by the omniduino omnibus to the library file directory compiled by

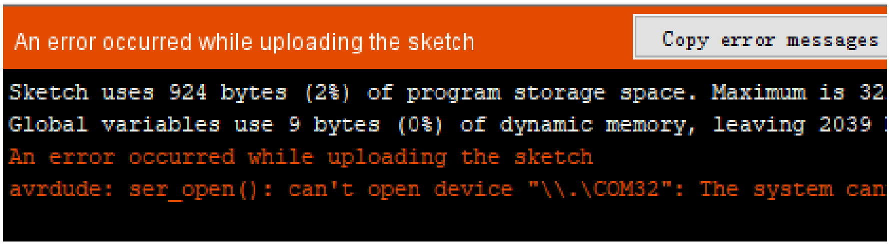
arduinoIDE.
please refer to 【3.Development Environment Construction】---- 【3.4 Add additional library files】



Adafruit_PWMServoDriver.h: No such file or directory

```
CarRun:2:10: error: Adafruit_PWMServoDriver.h: No such file or directory

 #include <Adafruit_PWMServoDriver.h>

          ^~~~~~~~~~~~~~~~~~~~~~~~~~~

compilation terminated.

exit status 1
Adafruit_PWMServoDriver.h: No such file or directory
```

5.3 If the compilation passes normally, but the following error occurs during uploading, the reason may be that the wrong serial port or the serial port is occupied.

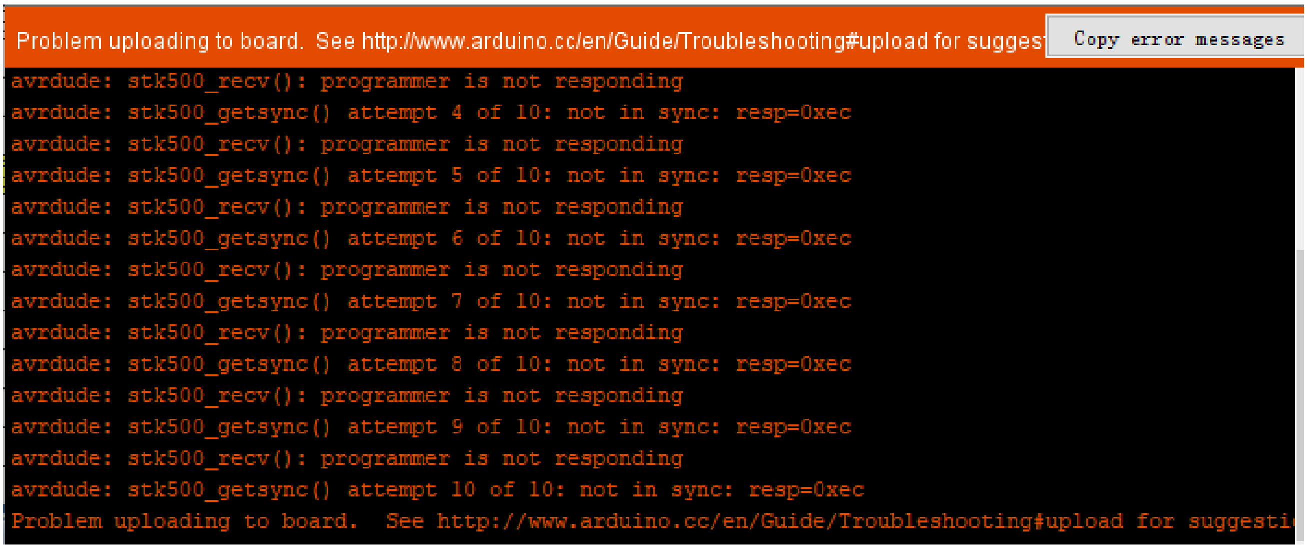


An error occurred while uploading the sketch

```
Sketch uses 924 bytes (2%) of program storage space. Maximum is 32
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 
An error occurred while uploading the sketch
avrdude: ser_open(): can't open device "\\.\COM32": The system can
```

Solution: Open the device manager to see if there is a serial port with CH340 tag. If not, please restart the Omniduino car, then, re-plug the USB cable or replace a USB cable;
If there is a serial port number, we need to close the other serial port or assistant software, avoid serial port occupation, and then re-select the serial port to ArduinoIDE【Tool】-->【Port】.

5.4 After clicking the upload button, the upload is always displayed, but it can't be uploaded successfully for a long time.



Problem uploading to board. See http://www.arduino.cc/en/Guide/Troubleshooting#upload for sugges

```
avrdude: stk500_recv(): programmer is not responding
avrdude: stk500_getsync() attempt 4 of 10: not in sync: resp=0xec
avrdude: stk500_recv(): programmer is not responding
avrdude: stk500_getsync() attempt 5 of 10: not in sync: resp=0xec
avrdude: stk500_recv(): programmer is not responding
avrdude: stk500_getsync() attempt 6 of 10: not in sync: resp=0xec
avrdude: stk500_recv(): programmer is not responding
avrdude: stk500_getsync() attempt 7 of 10: not in sync: resp=0xec
avrdude: stk500_recv(): programmer is not responding
avrdude: stk500_getsync() attempt 8 of 10: not in sync: resp=0xec
avrdude: stk500_recv(): programmer is not responding
avrdude: stk500_getsync() attempt 9 of 10: not in sync: resp=0xec
avrdude: stk500_recv(): programmer is not responding
avrdude: stk500_getsync() attempt 10 of 10: not in sync: resp=0xec
Problem uploading to board.  See http://www.arduino.cc/en/Guide/Troubleshooting#upload for suggesti
```

Because the uploading program and the WIFI camera communication is realized through the serial port, when the serial port is occupied by the WIFI camera, and the program cannot be uploaded.

Solution:

①Unplug the USB cable, turn off the power of the car, wait for the D2 indicator to go out.

②Then, plug in the USB data cable. At this time, your mobile phone should not connect the WiFi signal of the car.

③You can upload the program to the car according to the normal steps.

④After the program is successfully uploaded, unplug the USB data cable, open the power switch of the car. The corresponding experimental phenomenon will appear.

(Tip: If you upload APP control program. After the program is successfully uploaded, unplug the USB data cable, open the power switch of the car. Mobile phone connect the car to the WIFI signal, and then open the APP to control.)