# Network acquisition time

Difficult Level: C:\Users\52pidev1\AppData\Roaming\Tencent\TIM\Temp\T_ZKW6KJ_X{%[%P_AY$`(]X.png C:\Users\52pidev1\AppData\Roaming\Tencent\TIM\Temp\T_ZKW6KJ_X{%[%P_AY$`(]X.png C:\Users\52pidev1\AppData\Roaming\Tencent\TIM\Temp\T_ZKW6KJ_X{%[%P_AY$`(]X.png

## A. Get the time via the network and play the buzzer

### What you need

* 1 x Buzzer
* 1 x Arduino UNO R4 Wi-Fi
* 1 x USB 2.0 cable Type-C
* 2 x Male to male dumper wires

### How to wiring circuit diagram?

|  |  |
| --- | --- |
| Arduino UNO R4 WIFI | Buzzer |
| D8 | VCC(the longer pin) |
| GND | GND |

Then copy the code and upload it.

**// Include the RTC library**

**#include "RTC.h"**

**//Include the NTP library**

**#include <NTPClient.h>**

**#if defined(ARDUINO\_PORTENTA\_C33)**

**#include <WiFiC3.h>**

**#elif defined(ARDUINO\_UNOWIFIR4)**

**#include <WiFiS3.h>**

**#endif**

**#include <WiFiUdp.h>**

**#include "arduino\_secrets.h"**

**///////please enter your sensitive data in the Secret tab/arduino\_secrets.h**

**char ssid[] = SECRET\_SSID;        // your network SSID (name)**

**char pass[] = SECRET\_PASS;    // your network password (use for WPA, or use as key for WEP)**

**int wifiStatus = WL\_IDLE\_STATUS;**

**int buzzer = 8;**

**WiFiUDP Udp; // A UDP instance to let us send and receive packets over UDP**

**NTPClient timeClient(Udp);**

**void printWifiStatus() {**

**// print the SSID of the network you're attached to:**

**Serial.print("SSID: ");**

**Serial.println(WiFi.SSID());**

**// print your board's IP address:**

**IPAddress ip = WiFi.localIP();**

**Serial.print("IP Address: ");**

**Serial.println(ip);**

**// print the received signal strength:**

**long rssi = WiFi.RSSI();**

**Serial.print("signal strength (RSSI):");**

**Serial.print(rssi);**

**Serial.println(" dBm");**

**}**

**void connectToWiFi(){**

**// check for the WiFi module:**

**if (WiFi.status() == WL\_NO\_MODULE) {**

**Serial.println("Communication with WiFi module failed!");**

**// don't continue**

**while (true);**

**}**

**String fv = WiFi.firmwareVersion();**

**if (fv < WIFI\_FIRMWARE\_LATEST\_VERSION) {**

**Serial.println("Please upgrade the firmware");**

**}**

**// attempt to connect to WiFi network:**

**while (wifiStatus != WL\_CONNECTED) {**

**Serial.print("Attempting to connect to SSID: ");**

**Serial.println(ssid);**

**// Connect to WPA/WPA2 network. Change this line if using open or WEP network:**

**wifiStatus = WiFi.begin(ssid, pass);**

**// wait 10 seconds for connection:**

**delay(10000);**

**}**

**Serial.println("Connected to WiFi");**

**printWifiStatus();**

**}**

**void setup(){**

**Serial.begin(9600);**

**while (!Serial);**

**pinMode(buzzer, OUTPUT);**

**connectToWiFi();**

**RTC.begin();**

**Serial.println("\nStarting connection to server...");**

**timeClient.begin();**

**timeClient.update();**

**delay(2000);//It takes some time to get the time through the network, so we establish a delay**

**// Get the current date and time from an NTP server and convert**

**// it to CN +8 by passing the time zone offset in hours.**

**// You may change the time zone offset to your local one.**

**auto timeZoneOffsetHours = 8;**

**auto unixTime = timeClient.getEpochTime() + (timeZoneOffsetHours \* 3600);**

**Serial.print("Unix time = ");**

**Serial.println(unixTime);**

**RTCTime timeToSet = RTCTime(unixTime);**

**RTC.setTime(timeToSet);**

**// Trigger the alarm every time the minite are zero**

**RTCTime alarmTime;**

**alarmTime.setMinute(0);**

**// Make sure to only match on the minite in this example - not on any other parts of the date/time**

**AlarmMatch matchTime;**

**matchTime.addMatchMinute();**

**//sets the alarm callback**

**RTC.setAlarmCallback(alarmCallback, alarmTime, matchTime);**

**}**

**void loop(){**

**delay(1000);//ever 1s loop run once**

**// Retrieve the date and time from the RTC and print them**

**RTCTime currentTime;**

**RTC.getTime(currentTime);**

**Serial.println("The RTC was just set to: " + String(currentTime));**

**}**

**void alarmCallback(){**

**digitalWrite(buzzer, HIGH); //Make a sound**

**delay(3000);//delay 3000ms**

**digitalWrite(buzzer, LOW); //make the sound off**

**delay(300);//delay 300**

**}**

The following code shows how this code fetches the timestamp over the network, and fetches the time as you change the time zone from place to place.

**// Get the current date and time from an NTP server and convert**

**// it to UTC +8 by passing the time zone offset in hours.**

**// You may change the time zone offset to your local one.**

**auto timeZoneOffsetHours = 8;**

**auto unixTime = timeClient.getEpochTime() +**

**(timeZoneOffsetHours \* 3600);**

**Serial.print("Unix time = ");**

**Serial.println(unixTime);**

**RTCTime timeToSet = RTCTime(unixTime);**

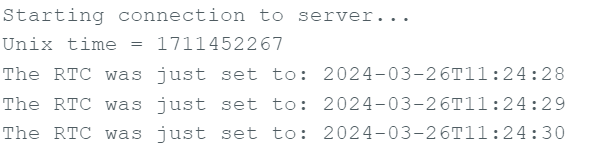
**RTC.setTime(timeToSet);**

**RTCTime currentTime;**

**RTC.getTime(currentTime);**

**Serial.println("The RTC was just set to: " + String(currentTime));**

And this is the output in serial monitor.



This is to the set alarm, the following is set for the 0th minute of each hour, to check whether it is the minute data, when the time is up, run the alarmCallback () function.

**// Trigger the alarm every time the minute are zero**

**RTCTime alarmTime;**

**alarmTime.setMinute(0);**

**// Make sure to only match on the minute in this example - not on any other parts of the date/time**

**AlarmMatch matchTime;**

**matchTime.addMatchMinute();**

**//sets the alarm callback**

**RTC.setAlarmCallback(alarmCallback, alarmTime, matchTime);**

**void alarmCallback(){**

**digitalWrite(buzzer, HIGH); //Make a sound**

**delay(3000);//delay 3000ms**

**digitalWrite(buzzer, LOW); //make the sound off**

**delay(300);//delay 300**

**}**

In the loop function, the time data is read every second and output to the serial monitor.

**void loop(){**

**delay(1000);//ever 1s loop run once**

**// Retrieve the date and time from the RTC and print them**

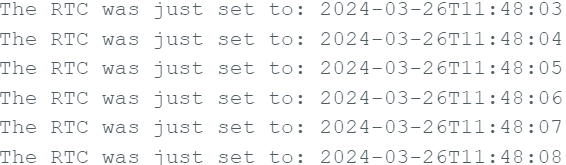
**RTCTime currentTime;**

**RTC.getTime(currentTime);**

**Serial.println("The RTC was just set to: " + String(currentTime));**

**}**

This is the data of the serial port monitor. You can open the serial monitor to see the date time.



## B. Extension

Get the time through the network, built-in simple music sequence, at a specific time when the alarm clock plays, and display the time to the LCD1602.

### What you need

* 1 x Buzzer
* 1 x Arduino UNO R4 Wi-Fi
* 1 x USB 2.0 cable Type C
* 2 x Male to male dumper wires
* 4 x Male to female dumper wires
* 1 x LCD(I2C)
* 1 x 52Pi Bread Board

### How to wiring circuit diagram?

|  |  |
| --- | --- |
| Arduino UNO R4 WIFI | Buzzer |
| D8 | VCC(the longer pin) |
| GND | GND |

|  |  |
| --- | --- |
| Arduino UNO R4 WIFI | I2C LCD 1602 |
| 5V | VCC |
| GND | GND |
| A4 | SDA |
| A5 | SCL |

Then upload the code:

**// Include the RTC library**

**#include "RTC.h"**

**//Include the NTP library**

**#include <NTPClient.h>**

**#if defined(ARDUINO\_PORTENTA\_C33)**

**#include <WiFiC3.h>**

**#elif defined(ARDUINO\_UNOWIFIR4)**

**#include <WiFiS3.h>**

**#endif**

**#include <WiFiUdp.h>**

**#include "arduino\_secrets.h"**

**#include <LiquidCrystal\_I2C.h>**

**LiquidCrystal\_I2C lcd(0x27, 16, 2); // I2C address 0x27, 16 column and 2 rows**

**int length = 28; // the number of  notes**

**char notes[] = "GGAGcB GGAGdc GGxecBA yyecdc";**

**int beats[] = {2,2,8,8,8,16,1,2,2,8,8,8,16,1,2,2,8,8,8,8,16,1,2,2,8,8,8,16};**

**int  tempo = 200;// time delay between notes**

**///////please enter your sensitive data in the Secret tab/arduino\_secrets.h**

**char ssid[] = SECRET\_SSID;        // your network SSID (name)**

**char pass[] = SECRET\_PASS;    // your network password (use for WPA, or use as key for WEP)**

**int wifiStatus = WL\_IDLE\_STATUS;**

**int buzzer = 8;**

**WiFiUDP Udp; // A UDP instance to let us send and receive packets over UDP**

**NTPClient timeClient(Udp);**

**void printWifiStatus() {**

**// print the SSID of the network you're attached to:**

**Serial.print("SSID: ");**

**Serial.println(WiFi.SSID());**

**// print your board's IP address:**

**IPAddress ip = WiFi.localIP();**

**Serial.print("IP Address: ");**

**Serial.println(ip);**

**// print the received signal strength:**

**long rssi = WiFi.RSSI();**

**Serial.print("signal strength (RSSI):");**

**Serial.print(rssi);**

**Serial.println(" dBm");**

**}**

**void connectToWiFi(){**

**// check for the WiFi module:**

**if (WiFi.status() == WL\_NO\_MODULE) {**

**Serial.println("Communication with WiFi module failed!");**

**// don't continue**

**while (true);**

**}**

**String fv = WiFi.firmwareVersion();**

**if (fv < WIFI\_FIRMWARE\_LATEST\_VERSION) {**

**Serial.println("Please upgrade the firmware");**

**}**

**// attempt to connect to WiFi network:**

**while (wifiStatus != WL\_CONNECTED) {**

**Serial.print("Attempting to connect to SSID: ");**

**Serial.println(ssid);**

**// Connect to WPA/WPA2 network. Change this line if using open or WEP network:**

**wifiStatus = WiFi.begin(ssid, pass);**

**// wait 10 seconds for connection:**

**delay(10000);**

**}**

**Serial.println("Connected to WiFi");**

**printWifiStatus();**

**}**

**void setup(){**

**Serial.begin(9600);**

**while (!Serial);**

**pinMode(buzzer, OUTPUT);**

**lcd.init();        // initialize the lcd**

**lcd.clear();       // clear the LCD display**

**lcd.backlight();   // Make sure backlight is on**

**connectToWiFi();**

**RTC.begin();**

**Serial.println("\nStarting connection to server...");**

**timeClient.begin();**

**timeClient.update();**

**// Get the current date and time from an NTP server and convert**

**// it to UTC +8 by passing the time zone offset in hours.**

**// You may change the time zone offset to your local one.**

**auto timeZoneOffsetHours = 8;**

**auto unixTime = timeClient.getEpochTime() + (timeZoneOffsetHours \* 3600);**

**Serial.print("Unix time = ");**

**Serial.println(unixTime);**

**RTCTime timeToSet = RTCTime(unixTime);**

**RTC.setTime(timeToSet);**

**// Trigger the alarm every time the minute are zero**

**RTCTime alarmTime;**

**alarmTime.setMinute(0);**

**// Make sure to only match on the minute in this example - not on any other parts of the date/time**

**AlarmMatch matchTime;**

**matchTime.addMatchMinute();**

**//sets the alarm callback**

**RTC.setAlarmCallback(alarmCallback, alarmTime, matchTime);**

**}**

**void playTone(int tone, int duration)  {**

**for (long i = 0; i < duration \* 1000L; i += tone \* 2) {**

**digitalWrite(buzzer,  HIGH);**

**delayMicroseconds(tone);**

**digitalWrite(buzzer, LOW);**

**delayMicroseconds(tone);**

**}**

**}**

**void playNote(char note, int duration) {**

**char names[] = {'C', 'D', 'E', 'F', 'G',  'A', 'B',**

**'c', 'd', 'e', 'f', 'g', 'a', 'b',**

**'x', 'y' };**

**int tones[] = { 1915, 1700, 1519, 1432, 1275,  1136, 1014,**

**956,  834,  765,  593,  468,  346,  224,**

**655 , 715 };**

**int SPEE = 5;**

**// play the tone corresponding  to the note name**

**for (int i = 0; i < 17; i++) {**

**if (names[i] ==  note) {**

**int newduration = duration/SPEE;**

**playTone(tones[i], newduration);**

**}**

**}**

**}**

**void loop(){**

**delay(1000);//ever 1s loop run once**

**// Retrieve the date and time from the RTC and print them**

**RTCTime currentTime;**

**RTC.getTime(currentTime);**

**Serial.println("The RTC was just set to: " + String(currentTime));**

**lcd.setCursor(0, 0);  // start to print at the first row**

**lcd.print(currentTime.getYear());**

**lcd.print("-");**

**lcd.print(Month2int(currentTime.getMonth()));**

**lcd.print("-");**

**lcd.print(currentTime.getDayOfMonth());**

**lcd.setCursor(0, 1);   // start to print at the second row**

**lcd.print(currentTime.getHour());**

**lcd.print(":");**

**lcd.print(currentTime.getMinutes());**

**lcd.print(":");**

**lcd.print(currentTime.getSeconds());**

**}**

**void alarmCallback(){**

**for (int i = 0; i < length; i++) {**

**if (notes[i] == ' ') {**

**delay(beats[i] \* tempo); // delay between notes**

**} else {**

**playNote(notes[i],   beats[i] \* tempo);**

**}**

**// time delay between notes**

**delay(tempo);**

**}**

**}**

Based on the previous project, the I2C library was added, various configurations of the songs played.

**#include <LiquidCrystal\_I2C.h>**

**LiquidCrystal\_I2C lcd(0x27, 16, 2); // I2C address 0x27, 16 column and 2 rows**

**int length = 28; // the number of  notes**

**char notes[] = "GGAGcB GGAGdc GGxecBA yyecdc";**

**int beats[] = {2,2,8,8,8,16,1,2,2,8,8,8,16,1,2,2,8,8,8,8,16,1,2,2,8,8,8,16};**

**int  tempo = 200;// time delay between notes**

The LCD is initialized in the setup function。And add two functions about tone and note.

Finally, in the loop function added to display the time to the LCD, in the timing callback function to play the song.

**lcd.setCursor(0, 0);  // start to print at the first row**

**lcd.print(currentTime.getYear());**

**lcd.print("-");**

**lcd.print(Month2int(currentTime.getMonth()));**

**lcd.print("-");**

**lcd.print(currentTime.getDayOfMonth());**

**lcd.setCursor(0, 1);   // start to print at the second row**

**lcd.print(currentTime.getHour());**

**lcd.print(":");**

**lcd.print(currentTime.getMinutes());**

**lcd.print(":");**

**lcd.print(currentTime.getSeconds());**

**}**

**void alarmCallback(){**

**for (int i = 0; i < length; i++) {**

**if (notes[i] == ' ') {**

**delay(beats[i] \* tempo); // delay between notes**

**} else {**

**playNote(notes[i],   beats[i] \* tempo);**

**}**

**// time delay between notes**

**delay(tempo);**

**}**

**}**

You can then change the time the alarm is set to make sure it rings. It rings every hour.

**// Trigger the alarm every time the minute are zero**

**RTCTime alarmTime;**

**alarmTime.setMinute(0);**

**// Make sure to only match on the minute in this example - not on any other parts of the date/time**

**AlarmMatch matchTime;**

**matchTime.addMatchMinute();.**

**//sets the alarm callback**

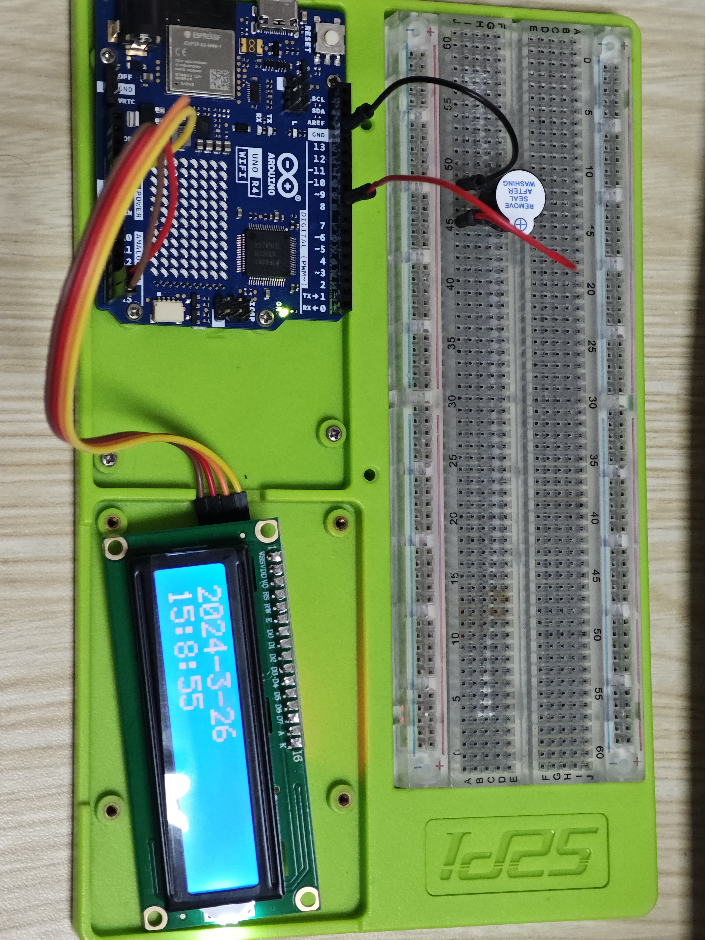
**RTC.setAlarmCallback(alarmCallback, alarmTime, matchTime);**

You can also access other objects in RTCTime by changing the function you call:

setHour () /\* from 0 (midnight) to 23 \*/ //alarmTime.setHour (8) rings at every 8 am

addMatchHour () //match if it’s hours or not

They make your alarm go off when you want it to go off.



This is the finally result. You can also display a few other things on the LCD to make it look cleaner.

## C. Extension by Speaker Amplifier and 3Ω Speaker module

With the kit inside the module to achieve network access to time, and timing alarm clock to play music

### What you need

* 1 x Speaker Amplifier
* 1 x 3Ω Speaker Module
* 1 x Arduino UNO R4 Wi-Fi
* 1 x USB 2.0 cable Type C
* 6 x male to male dumper wires

### How to wiring circuit diagram?

|  |  |
| --- | --- |
| Arduino UNO R4 WIFI | Speaker Amplifier |
| 5V | VCC |
| A0 | Audio input |
| GND | AGND |
| GND | MUTE |

|  |  |
| --- | --- |
| Speaker Amplifier | 3Ω Speaker Module |
| Audio output+ | VCC(the Red pin) |
| Audio output- | GND(the black) |

Copy the code and paste it to the IDE.

**// Include the RTC library**

**#include "RTC.h"**

**//Include the NTP library**

**#include <NTPClient.h>**

**#include <WiFiS3.h>**

**#include <WiFiUdp.h>**

**#include "arduino\_secrets.h"**

**#include "analogWave.h"**

**analogWave wave(DAC);**

**#define NOTE\_A4 69**

**#define FREQ\_A4 440**

**int tonic = 65;**

**int melody[] = {**

**1,3,5,1,**

**1,3,5,1,**

**5,6,8,**

**5,6,8,**

**8,10,8,6,5,1,**

**8,10,8,6,5,1,**

**1, -4, 1,**

**1, -4, 1,**

**};**

**int rhythm[] {**

**4, 4, 4, 4,**

**4, 4, 4, 4,**

**4, 4, 2,**

**4, 4, 2,**

**8, 8, 8, 8, 4, 4,**

**8, 8, 8, 8, 4, 4,**

**4, 4, 2,**

**4, 4, 2**

**};**

**int noteCounter = 0;**

**int bpm = 120;**

**float beatDuration = 60.0 / bpm \* 1000;**

**///////please enter your sensitive data in the Secret tab/arduino\_secrets.h**

**char ssid[] = SECRET\_SSID;        // your network SSID (name)**

**char pass[] = SECRET\_PASS;    // your network password (use for WPA, or use as key for WEP)**

**int wifiStatus = WL\_IDLE\_STATUS;**

**WiFiUDP Udp; // A UDP instance to let us send and receive packets over UDP**

**NTPClient timeClient(Udp);**

**int flag=64;**

**void alarmCallback(){**

**while(flag){**

**int currentNote = melody[noteCounter] + tonic;**

**float frequncy = FREQ\_A4 \* pow(2, ((currentNote - NOTE\_A4) / 12.0));**

**float noteDuration = beatDuration \* (4.0 / rhythm[noteCounter]);**

**wave.freq(frequncy);**

**delay(noteDuration \* 0.85);**

**wave.stop();**

**delay(noteDuration \* 0.15);**

**noteCounter++;**

**noteCounter = noteCounter % 32;**

**flag--;**

**delay(1);**

**if(flag<=1)wave.stop();**

**}**

**}**

**void printWifiStatus() {**

**// print the SSID of the network you're attached to:**

**Serial.print("SSID: ");**

**Serial.println(WiFi.SSID());**

**// print your board's IP address:**

**IPAddress ip = WiFi.localIP();**

**Serial.print("IP Address: ");**

**Serial.println(ip);**

**// print the received signal strength:**

**long rssi = WiFi.RSSI();**

**Serial.print("signal strength (RSSI):");**

**Serial.print(rssi);**

**Serial.println(" dBm");**

**}**

**void connectToWiFi(){**

**// check for the WiFi module:**

**if (WiFi.status() == WL\_NO\_MODULE) {**

**Serial.println("Communication with WiFi module failed!");**

**// don't continue**

**while (true);**

**}**

**String fv = WiFi.firmwareVersion();**

**if (fv < WIFI\_FIRMWARE\_LATEST\_VERSION) {**

**Serial.println("Please upgrade the firmware");**

**}**

**// attempt to connect to WiFi network:**

**while (wifiStatus != WL\_CONNECTED) {**

**Serial.print("Attempting to connect to SSID: ");**

**Serial.println(ssid);**

**// Connect to WPA/WPA2 network. Change this line if using open or WEP network:**

**wifiStatus = WiFi.begin(ssid, pass);**

**// wait 10 seconds for connection:**

**delay(10000);**

**}**

**Serial.println("Connected to WiFi");**

**printWifiStatus();**

**}**

**void setup(){**

**Serial.begin(9600);**

**while (!Serial);**

**wave.sine(10);**

**wave.stop();**

**connectToWiFi();**

**RTC.begin();**

**Serial.println("\nStarting connection to server...");**

**timeClient.begin();**

**timeClient.update();**

**// Get the current date and time from an NTP server and convert**

**// it to CN +8 by passing the time zone offset in hours.**

**// You may change the time zone offset to your local one.**

**auto timeZoneOffsetHours = 8;**

**auto unixTime = timeClient.getEpochTime() + (timeZoneOffsetHours \* 3600);**

**Serial.print("Unix time = ");**

**Serial.println(unixTime);**

**RTCTime timeToSet = RTCTime(unixTime);**

**RTC.setTime(timeToSet);**

**// Trigger the alarm every time the minite are zero**

**RTCTime alarmTime;**

**alarmTime.setMinute(0); /\* from 0 to 59 \*/**

**// Make sure to only match on the minite in this example - not on any other parts of the date/time**

**AlarmMatch matchTime;**

**matchTime.addMatchMinute();**

**//sets the alarm callback**

**RTC.setAlarmCallback(alarmCallback, alarmTime, matchTime);**

**}**

**void loop(){**

**delay(1000);//ever 1s loop run once**

**// Retrieve the date and time from the RTC and print them**

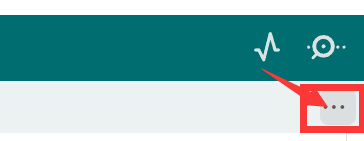
**RTCTime currentTime;**

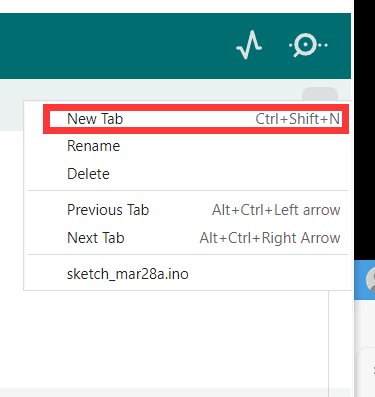
**RTC.getTime(currentTime);**

**Serial.println("The RTC was just set to: " + String(currentTime));**

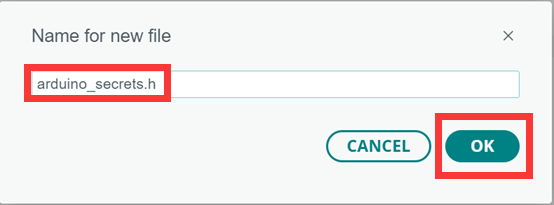
**}**

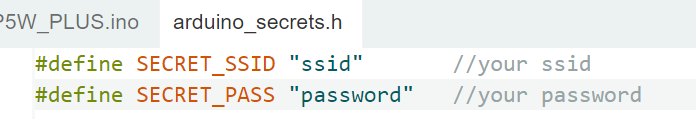
Then click this to add the .h file.





Then write down the arduino\_secrets.h and fill in your ssid and password.





This song is 32 beats in total, and 64 is used to make him play it twice.

**int flag=64;**

Then you can choose a song you like and search for its melody and beat and change it.

Finally, you can upload and run it, and it will ring when the time reaches the 0th minute of each hour. (Note: Sometimes the network affects the acquisition time. If you want it to ring faster, you can change the zero below to two minutes from your current time, and don't forget to change the time zone)

**alarmTime.setMinute(0); /\* from 0 to 59 \*/**

Have you implemented the function of this alarm clock? Go show off to your friends! Let’s get into next chapter!