

Lektion 3 Infrarot- Gesteuertes Auto



Die Punkte des Abschnitts

Infrarot- Fernbedienung ist eine weit verbreitete Methode für die Fernbedienungen. Das Auto wurde mit Infrarot- Empfänger ausgestattet und ermöglicht so die Steuerung über die Infrarot- Fernbedienung.

Lernteile:

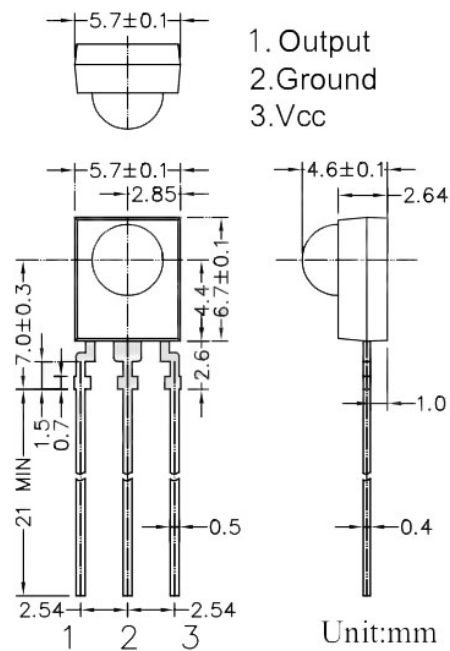
- ◆ Verstehen Sie die Infrarot- Fernbedienung und den Empfänger
- ◆ Verstehen Sie die Fernsteuerungsprinzipien

Vorbereitungen:

- ◆ Ein Auto (mit Batterie)
- ◆ Ein USB cable
- ◆ IR receiving module und IR Fernbedienung

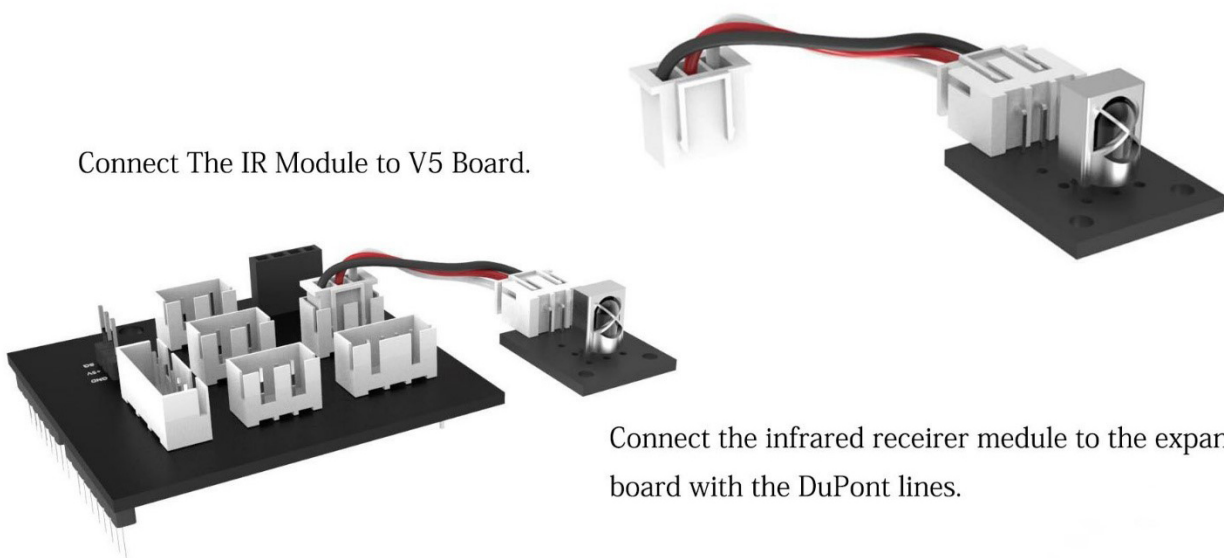
I . IR Empfangsmodul und IR- Fernbedienung

Die Daten des IR -Empfängersensors sind wie folgt:



Die Verbindung des Empfängermoduls ist wie folgt:

Connect The IR Module to V5 Board.



Connect the infrared receiver module to the expansion board with the DuPont lines.

Das ist die IR- Fernbedienung:



II. Test- Programm

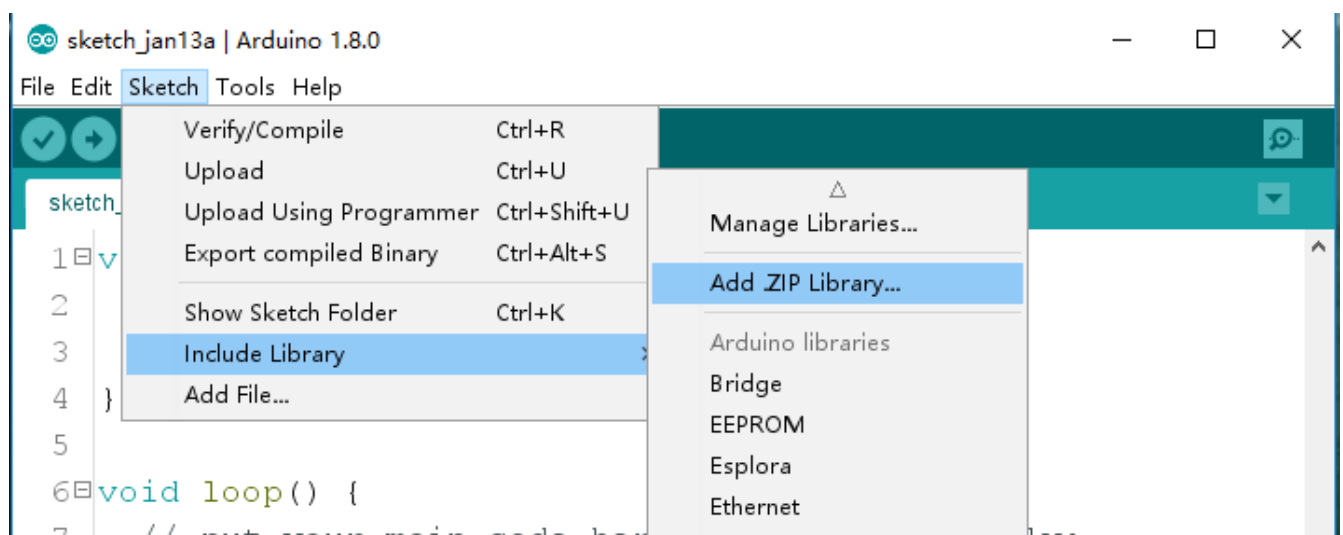
Da wir in diesem Programm eine zusätzliche Bibliothek benötigen, laden wir diese zu allererst in unser

Dateisystem.

Öffnen Sie die Arduino IDE



Klicken Sie auf Sketch- Include Library- Add .ZIP- Bibliothek ... - wählen Sie die Bibliothek wie unten aus.

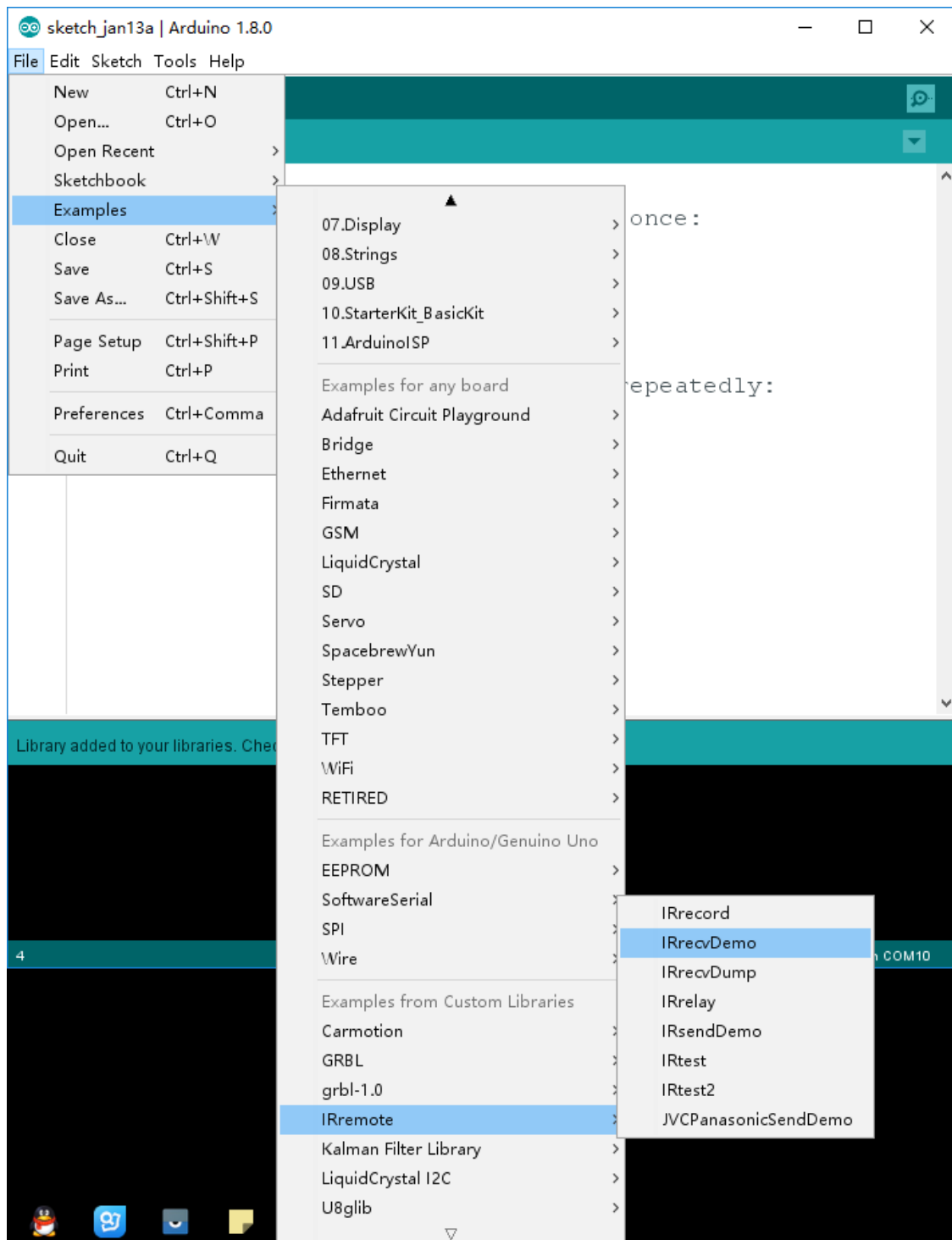


Der Dateiname der ZIP- Bibliothek muss IRremote.zip sein.

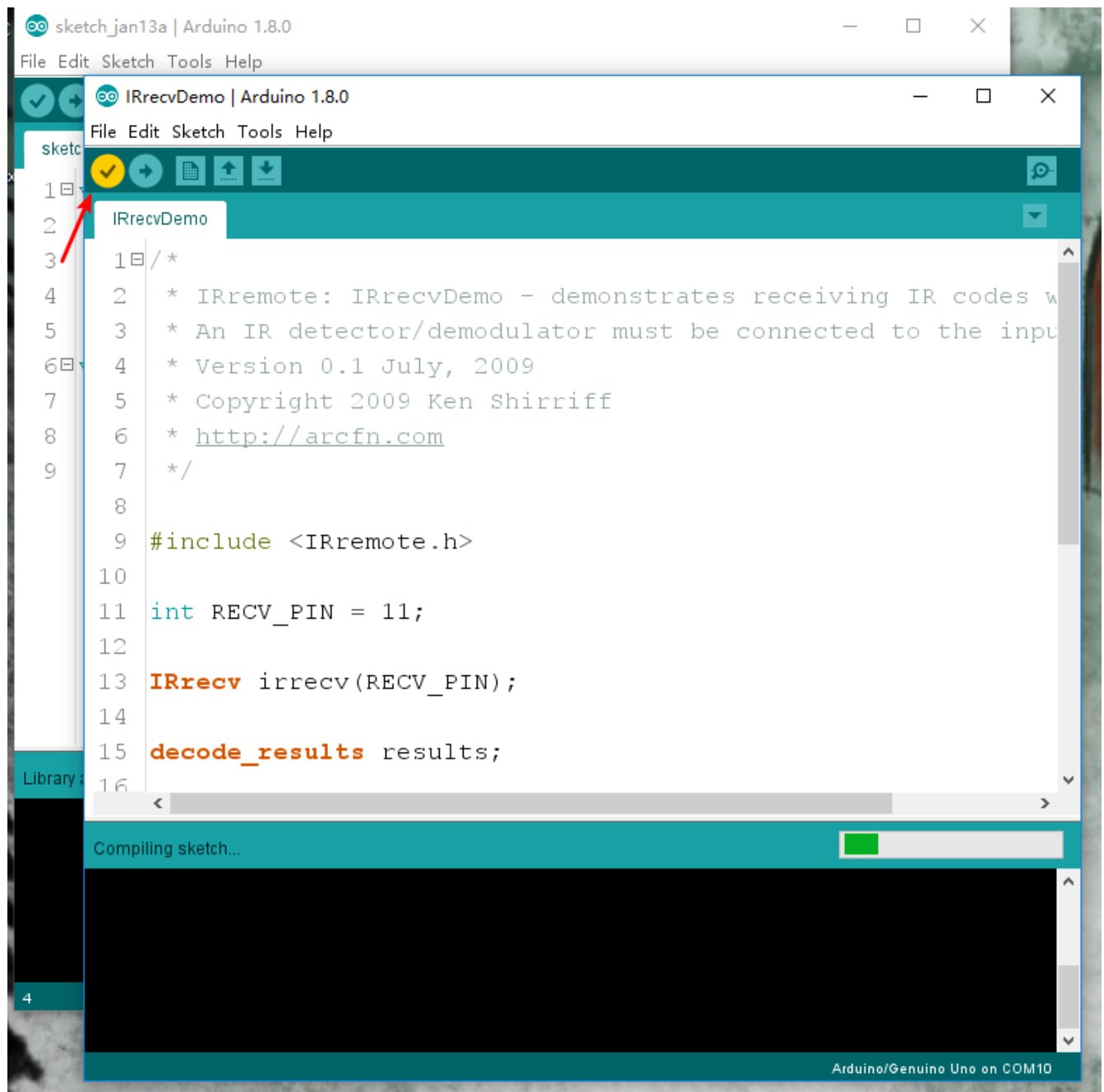
< < Elegoo Smart Robot Car Kit V2.0 > Lesson 3 Infrared Remote Control Car	
名称	修改日期
IRremote.zip	2017/1/11 15:29

Das Programm muss mit dieser Bibliotheksdatei kompiliert werden, da es eine speziell modifizierte Bibliotheksdatei ist.

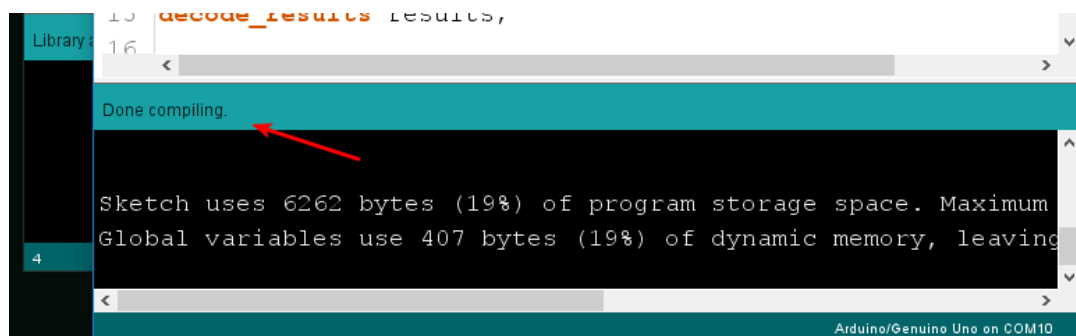
Wähle ein IRremote- Beispiel




Klicken Sie auf die Schaltfläche Kompilieren.



Fertig kompiliert. Wenn nicht, wurde die IRremote- Bibliothek nicht erfolgreich installiert. Bitte füge die IRremote- Bibliothek noch einmal hinzu.



Öffnen Sie die Datei infrared_Blink \ infrared_Blink.ino

< < Lesson 3 Infrared Remote Control Car > infrared_Blink		
名称	修改日期	类型
 infrared_Blink.ino	2017/1/5 11:57	Ardui

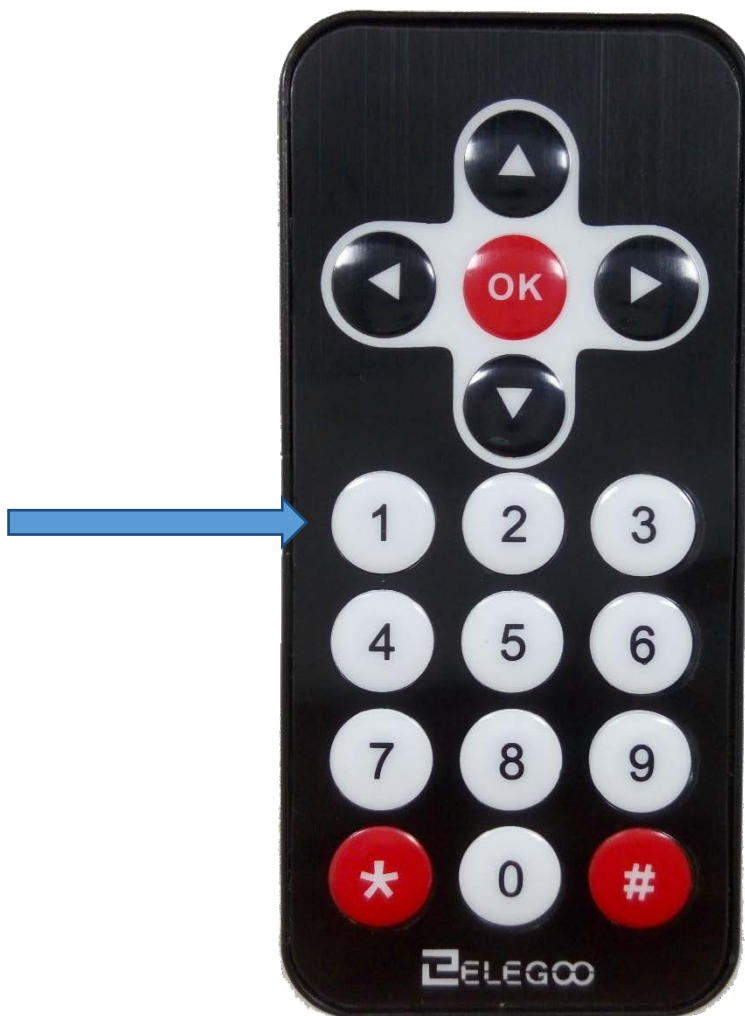
Der Code sieht folgendermaßen aus:

```
#include <IRremote.h> // IR Bibliothek einfügen
int receiverpin = 12; // Infrarot- Signalempfangspin
int LED=13; // definiert LED pin
volatile int state = LOW; // Definiert den Standard- Eingabemodus
unsigned long RED;
#define L 16738455
IRrecv irrecv(receiverpin); // initialization
decode_results results; // definiert den Strukturtyp
void setup() {
    pinMode(LED, OUTPUT); // Initialisierung der LED als Ausgang
    Serial.begin(9600); // Debug- Ausgabe bei 9600 Baud
    irrecv.enableIRIn(); // Startet den empfang
}
void stateChange()
{
    state = !state;
    digitalWrite(LED, state);
}
void loop() {
    if (irrecv.decode(&results))
    {
        RED=results.value;
        Serial.println(RED);
        irrecv.resume(); // Erhalten den nächsten Wert
        delay(150);
        if(RED==L)
```



```
{  
  stateChange();  
}  
}  
}
```

Laden Sie das Programm auf die ROMEO-Controller-Karte. Nach dem Trennen des Autos auf den Computer können Sie den Netzschalter einschalten und das Fahrzeug auf den Boden stellen. Drücken Sie die Taste "1" in Richtung des Autos, beobachten Sie das Auto, und Sie werden sehen, dass sich die LED ausschaltet.



III. Einführung des Grundprinzips

1. Arbeitsprinzip

Das Universal- Infrarot- Fernsteuerungssystem besteht aus zwei Teilen: Senden und Empfangen, das sendende Teil besteht aus einer IR-Fernbedienung, das empfangende Teil besteht aus einem Infrarot-Empfangsrohr. Die Signale, die durch IR- Fernsteuerung gesendet werden, sind ein serieller binärer Puls- Code. Um von der Ablenkung anderer Infrarotsignale während des drahtlosen Transports frei zu sein, ist es allgemein üblich, sie bei der gegebenen Trägerfrequenz zu modulieren und sie dann durch einen infrarot ausgestrahlten Phototransistor zu senden. Infrarot- Empfangsschlauch filtert andere Rauschwellen aus, empfängt nur Signale mit gegebener Frequenz und stellt dann den Binärpulscode mit Hilfe der Demodulation wieder her. Der Eingebaute Empfangsschlauch empfängt und verwandelt Lichtsignale, die von Infrarot-Leuchtdioden mit schwachen elektrischen Signalen gesendet werden. Die Signale werden durch Verstärker im IC verstärkt und durch automatische Verstärkungssteuerung, Bandpassfilterung, Demodulation, Wellenformung und so weiter wiederhergestellt. Die Codierung, die per Fernbedienung gesendet wird, erkennt die Schaltung durch die Codierung, die in das elektrische Gerät über den Signalausgangspin des Infrarot- Empfangsmoduls eingegeben wird.

2. Protokoll der Infrarotfernsteuerung

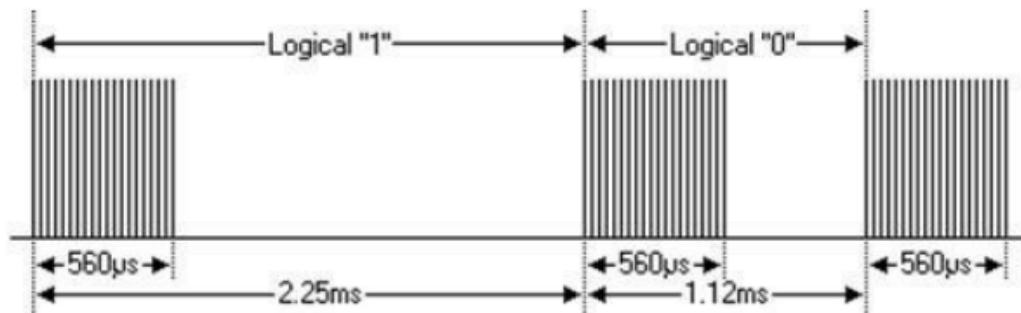
Das Codierungsschema der angepassten IR-Fernsteuerung ist: NEC-Protokoll.

Als nächstes wollen wir lernen, was NEC-Protokoll ist.

Eigenschaften:

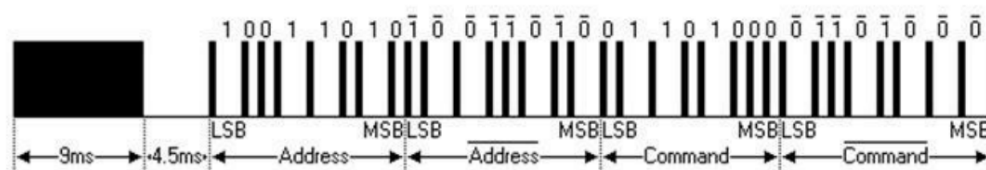
- (1) 8 Adreßbit, 8 Ordnung Bit
- (2) Adressbit und Bestellbit werden zweimal gesendet, um die Zuverlässigkeit zu gewährleisten
- (3) Pulspositionsmodulation
- (4) Trägerfrequenz ist 38kHz
- (5) Die Zeit eines jeden Bits beträgt 1.125ms oder 2.25ms

Definitionen von logischen 0 und 1 sind wie folgt:



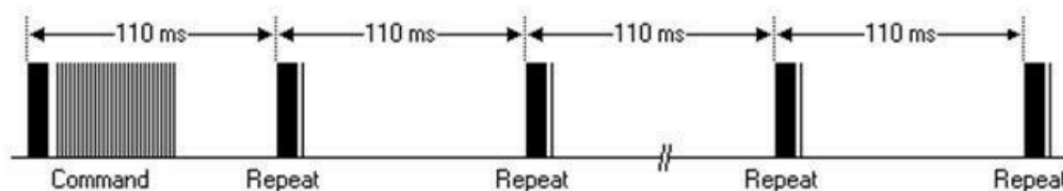
Das Protokoll ist wie folgt:

Drücken Sie sofort den Übertragungsimpuls:



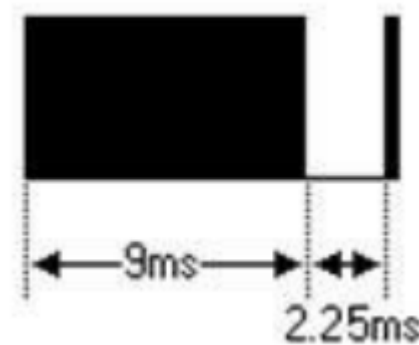
Hinweis: Dies ist das Protokoll des Sendens von LSB ((least-significant bit)am wenigsten signifikantes Bit). Transportadresse des obigen Impulses ist 0x59, Auftrag ist 0x16. Eine Nachricht beginnt von einem hohen Niveau von 9ms, die Folge ist ein niedriger Grad von 4,5ms, (zwei Ebenen Formular Leitfaden) und durch Adresscode und Bestellcode. Adresse und Bestellung werden zweimal übermittelt. Beim zweiten Mal können alle Bits umgekehrt umgekehrt werden, um die zu verwendenden Empfangsmeldungen zu bestätigen. Die Gesamtsendungszeit ist fest, wenn Sie nicht daran interessiert sind, können Sie die Zuverlässigkeit der Invertierung ignorieren und können Adresse und Ordnung um 16 bit erweitern! Weil die Tatsache, dass Länge wiederholen, wenn jedes Bit ist entgegengesetzt.

Drücken Sie den gesendeten Puls nach einiger Zeit.



Ein Befehl wird gesendet, auch wenn die Taste der Fernbedienung immernoch gedrückt wird. Wenn die Taste noch gedrückt wird, ist der Puls der ersten 110ms von oben verschieden, der doppelte Code wird nach allen 110ms übertragen. Der doppelte Code besteht aus einem Hochpegelimpuls von 9ms und einem niedrigen Niveau von 2,25 und einem hohen Niveau von 560µs.

Wiederholen Sie den Puls:



Anmerkung: Nachdem die Impulswellenform in die Integration des Sensors eintritt, aufgrund der Tatsache, dass die Integration des Sensors dekodiert, signalvergrößert und plastisch sein sollte, sollten Sie die Zeit beachten, in der keine Infrarotsignale vorhanden sind, deren Ausgangsklemme auf hohem Niveau ist, wenn es Signale gibt. So ist das Ausgangssignal dem Sendeterminal entgegengesetzt. Jeder kann sehen, Empfänger Puls durch Oszilloskop, verstehen Programm mit Wellenform gesehen.

3. Die Idee der Programmierung eines solchen Ferngesteuerten Autos

Entsprechend der Charakteristik des NEC-Codes und der Welle des Empfangs-Endes teilt dieses Experiment die Welle des Empfangs-Endes in vier Teile: Leitcode (Puls von 9ms und 4,5ms), Adresscode (einschließlich 8-Bit-Adresscode und 8-Bit) Adresse abrufen), 16-Bit-Adresscode (inkl. 8-Bit-Adresscode und 8-Bit-Adressabruf), 16-Bit-Bestellcode (inkl. 8-Bit-Bestellcode und 8-Bit-Bestellabruf), Wiederholungscode (bestehend aus Puls von 9ms) , 2,25 ms, 560us).

Nutzen Sie den Timer, um den hohen Pegel und den niedrigen Wellenpegel zu testen, der nach der getesteten Zeit unterschieden wird: logisch "01", logisch "1", führender Puls, Wiederholimpuls. Führender Code und Adresscode werden beurteilt ob sie korrekt sind, werden nicht gespeichert, aufgrund der Tatsache, dass der Bestellcode jeder Taste unterschiedlich ist, wird die Aktion durch den Bestellcode ausgeführt.

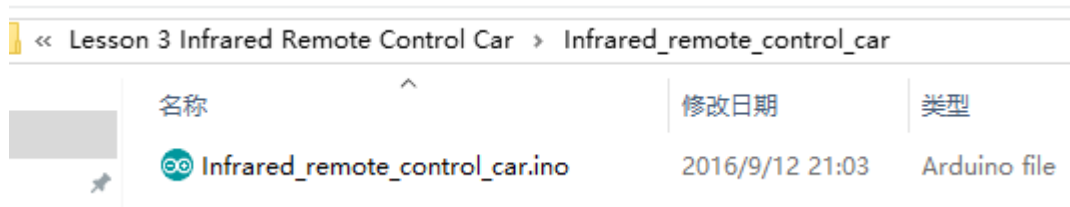
Während des Auto-Experiments müssen wir nur das Auto steuern, um vorwärts und rückwärts zu fahren, nach links und rechts zu drehen und zu stoppen, was bedeutet, dass wir 5 Schlüssel benötigen und der Wert von ihnen ist wie unten zu sehen:

Fernsteuerungszeichen	Schlüssel- Variable
Mittlerer roter Knopf	16712445
Oberes Dreieck	16736925
Unteres Dreieck	16754775
Linkes Dreieck	16720605
Rechtes Dreieck	16761405

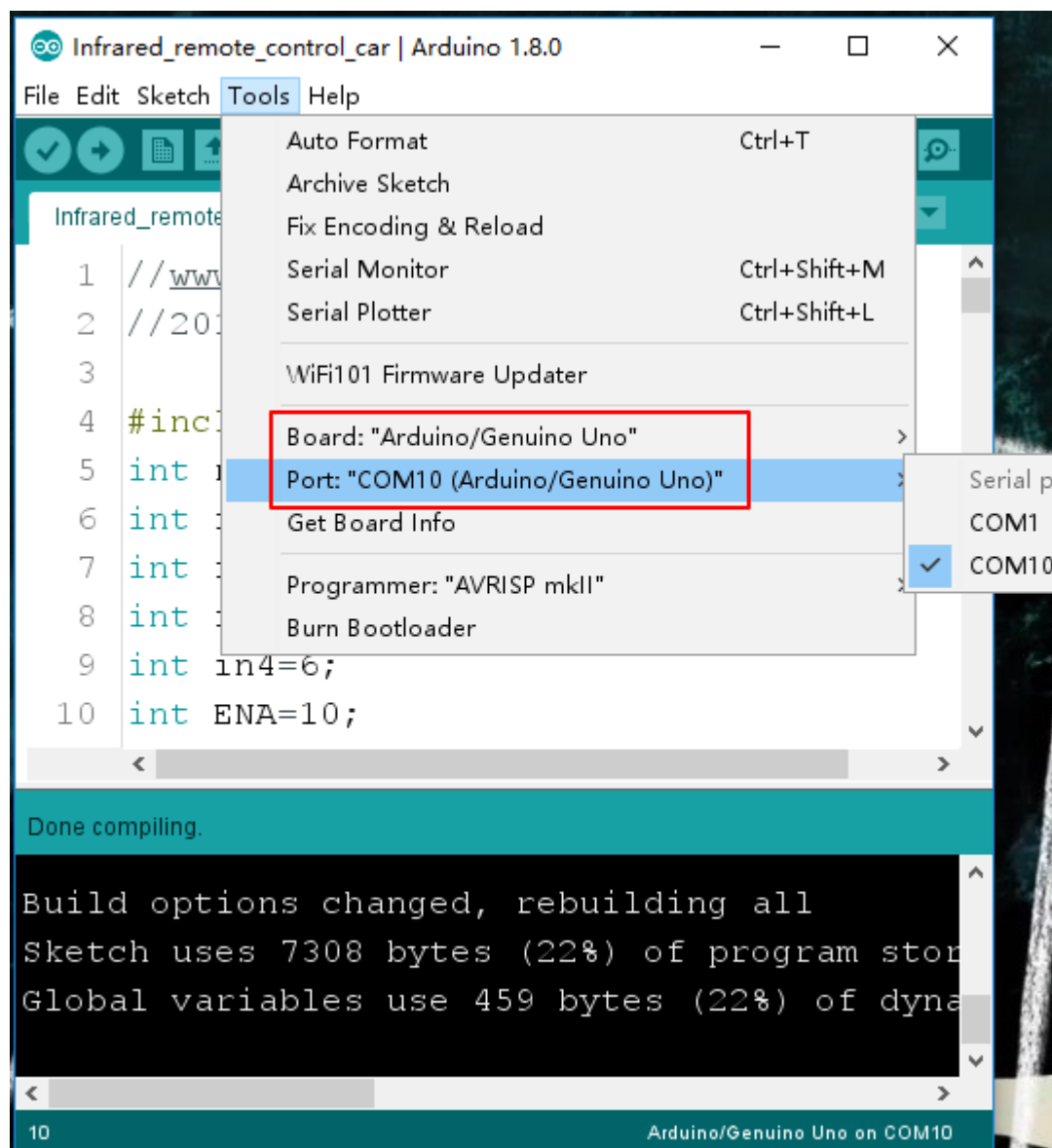


IV. Bau ein Ferngesteuertes Auto

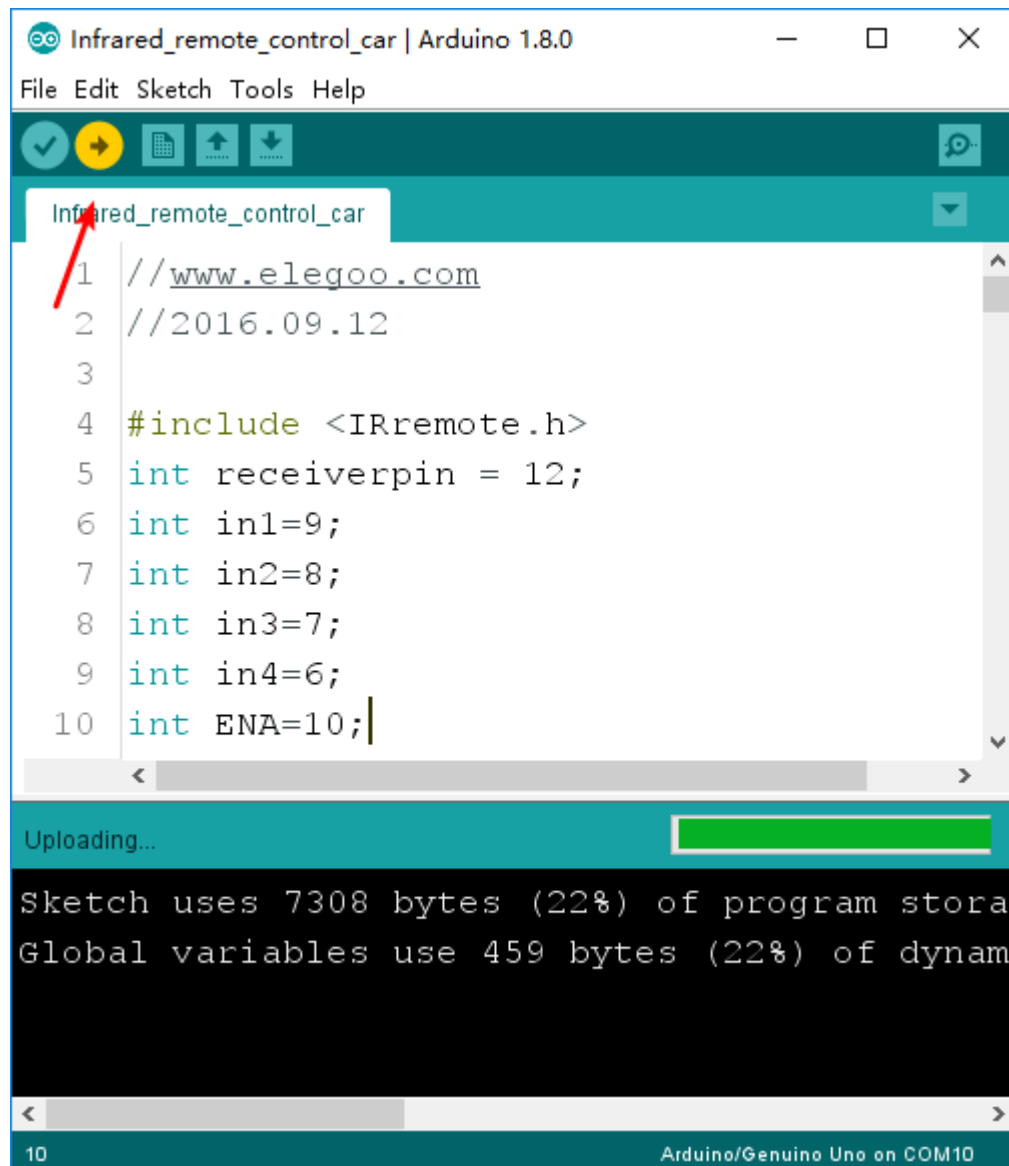
Wir laden das Programm wie unten auf das Auto, öffnen Sie die Datei Infrared_remote_control_car \ Infrared_remote_control_car.ino



Wählen Sie die Arduino Uno Board und Serial Port.



Drücken Sie die Upload-Taste



The screenshot shows the Arduino IDE interface. The title bar reads 'Infrared_remote_control_car | Arduino 1.8.0'. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. The toolbar contains icons for checking, uploading, creating a new sketch, opening a sketch, and saving a sketch. The file explorer shows the current sketch 'Infrared_remote_control_car'. The code editor displays the following code:

```

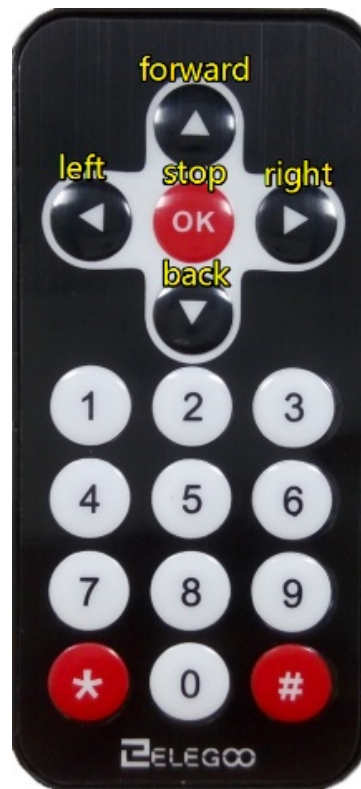
1 //www.elegoo.com
2 //2016.09.12
3
4 #include <IRremote.h>
5 int receiverpin = 12;
6 int in1=9;
7 int in2=8;
8 int in3=7;
9 int in4=6;
10 int ENA=10;
  
```

Below the code editor, a progress bar indicates the upload status. The status bar at the bottom shows '10' and 'Arduino/Genuino Uno on COM10'. The output window at the bottom displays the following message:

```

Sketch uses 7308 bytes (22%) of program storage
Global variables use 459 bytes (22%) of dynamic
  
```

Nach dem Hochladen, trennen Sie das Auto von den Computer. Dann schalten Sie den Netzschalter ein und stellen den Wagen auf den Boden. Drücken Sie die Taste auf der Fernbedienung und Sie können sehen, das Auto bewegen sich entsprechend, wie Sie befehlen.



Voila, jetzt kannst du mit dem IR-Steuerwagen glücklich spielen.

Der Code sieht wie folgt aus:

```
#include <IRremote.h>
int receiverpin = 12;
int in1=6;
int in2=7;
int in3=8;
int in4=9;
int ENA=5;
int ENB=11;
int ABS=150;
unsigned long RED;
#define A 16736925

#define B 16754775

#define X 16712445
```



```
#define C 16720605
```

```
#define D 16761405
```

```
IRrecv irrecv(receiverpin);
```

```
decode_results results;
```

```
void _mForward()
```

```
{
```

```
    digitalWrite(ENA,HIGH);
```

```
    digitalWrite(ENB,HIGH);
```

```
    digitalWrite(in1,HIGH);//digital output
```

```
    digitalWrite(in2,LOW);
```

```
    digitalWrite(in3,LOW);
```

```
    digitalWrite(in4,HIGH);
```

```
    Serial.println("go forward!");
```

```
}
```

```
void _mBack()
```

```
{
```

```
    digitalWrite(ENA,HIGH);
```

```
    digitalWrite(ENB,HIGH);
```

```
    digitalWrite(in1,LOW);
```

```
    digitalWrite(in2,HIGH);
```

```
    digitalWrite(in3,HIGH);
```

```
    digitalWrite(in4,LOW);
```

```
    Serial.println("go back!");
```

```
}
```

```
void _mleft()
```

```
{
```

```
    analogWrite(ENA,ABS);
```

```
    analogWrite(ENB,ABS);
```

```
    digitalWrite(in1,HIGH);
```

```
digitalWrite(in2,LOW);
digitalWrite(in3,HIGH);
digitalWrite(in4,LOW);
Serial.println("go left!");
}

void _mright()
{
  analogWrite(ENA,ABS);
  analogWrite(ENB,ABS);
  digitalWrite(in1,LOW);
  digitalWrite(in2,HIGH);
  digitalWrite(in3,LOW);
  digitalWrite(in4,HIGH);
  Serial.println("go right!");
}

void _mStop()
{
  digitalWrite(ENA,LOW);
  digitalWrite(ENB,LOW);
  Serial.println("STOP!");
}

void setup() {
  // put your setup code here, to run once:
  pinMode(in1,OUTPUT);
  pinMode(in2,OUTPUT);
  pinMode(in3,OUTPUT);
  pinMode(in4,OUTPUT);
  pinMode(ENA,OUTPUT);
  pinMode(ENB,OUTPUT);
  pinMode(receiverpin,INPUT);
  Serial.begin(9600);
  _mStop();
  irrecv.enableIRIn();
```

```
}

void loop() {
  if (irrecv.decode(&results))
  {

    RED=results.value;
    Serial.println(RED);
    irrecv.resume();
    delay(150);
    if(RED==A)
    {
      _mForward();
    }

    else if(RED==B)
    {
      _mBack();
    }

    else if(RED==C)
    {
      _mleft();
    }

    else if(RED==D)
    {
      _mright();
    }

    else if(RED==X)
    {
      _mStop();
    }
  }
}
```

}

}

}