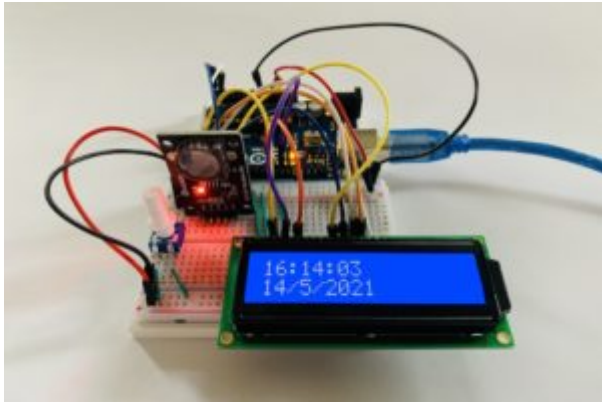


Die Uhrzeit mit einer RTC (Real Time Clock) ermitteln und anzeigen



Brauchst du für dein Arduino-Projekt die aktuelle Uhrzeit – vielleicht, weil du dir mit einem LCD-Display eine Uhr bauen möchtest, oder einen Zeitstempel für deine Daten benötigst? Dann eignet sich hierfür perfekt die RTC, oder Real Time Clock.

In diesem Tutorial erfährst du, wie du die das RTC-Modul DS1307 am Arduino anschließt und damit die aktuelle Uhrzeit samt Datum auf einem Display anzeigst.

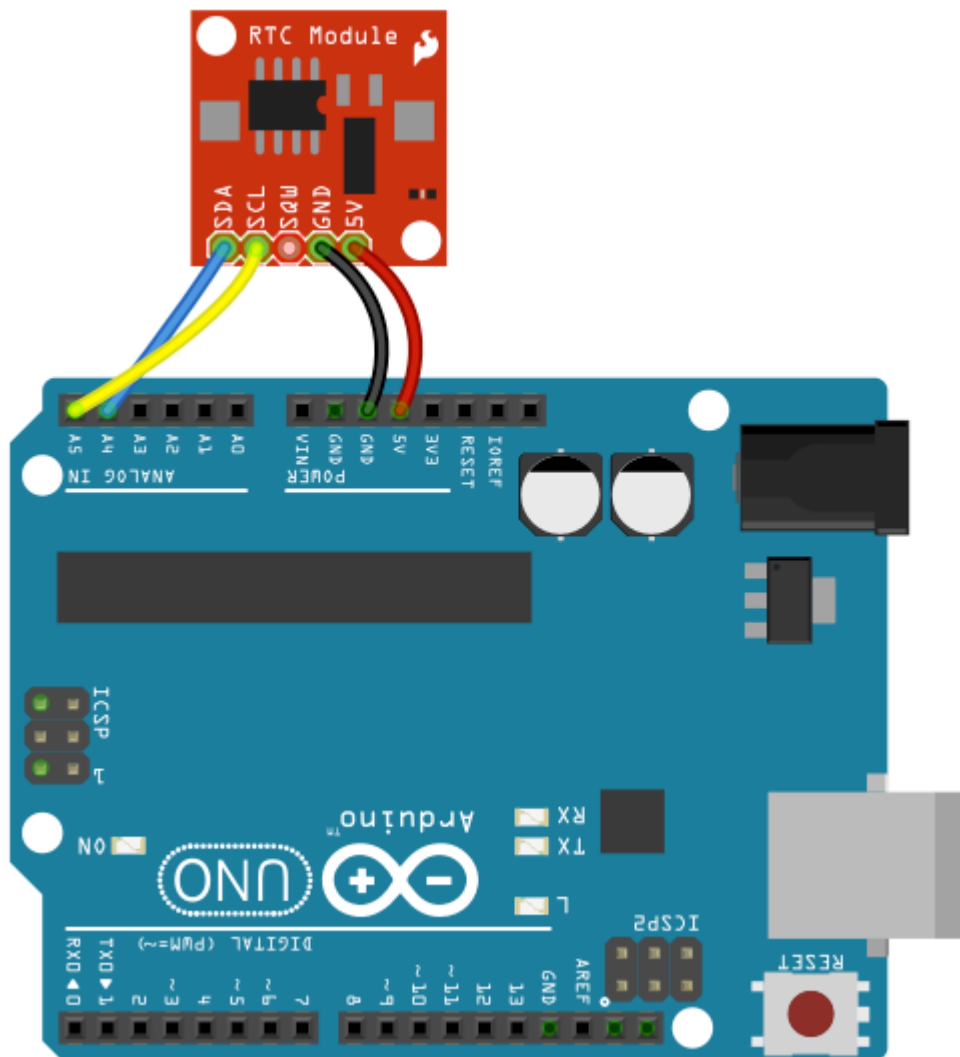
Was ist eine Real Time Clock?

In den meisten Fällen erhältst du beim Kauf einer [Real Time Clock*](#) ein Modul, auf dem oft ein DS1307 samt einer kleinen Batterie verbaut ist. Diese sorgt dafür, dass dein Arduino die aktuelle Zeit ermitteln kann, auch wenn es zwischenzeitlich zu einem Stromausfall gekommen ist – denn die RTC wird weiterhin mit Strom versorgt.

Diese Bauteile sind in der Regel recht preisgünstig und werden per I²C am Microcontroller angeschlossen. Die Genauigkeit ist für die allermeisten Projekte ausreichend und die Knopfbatterie hält bis zu fünf Jahre durch.

Die RTC am Arduino anschließen

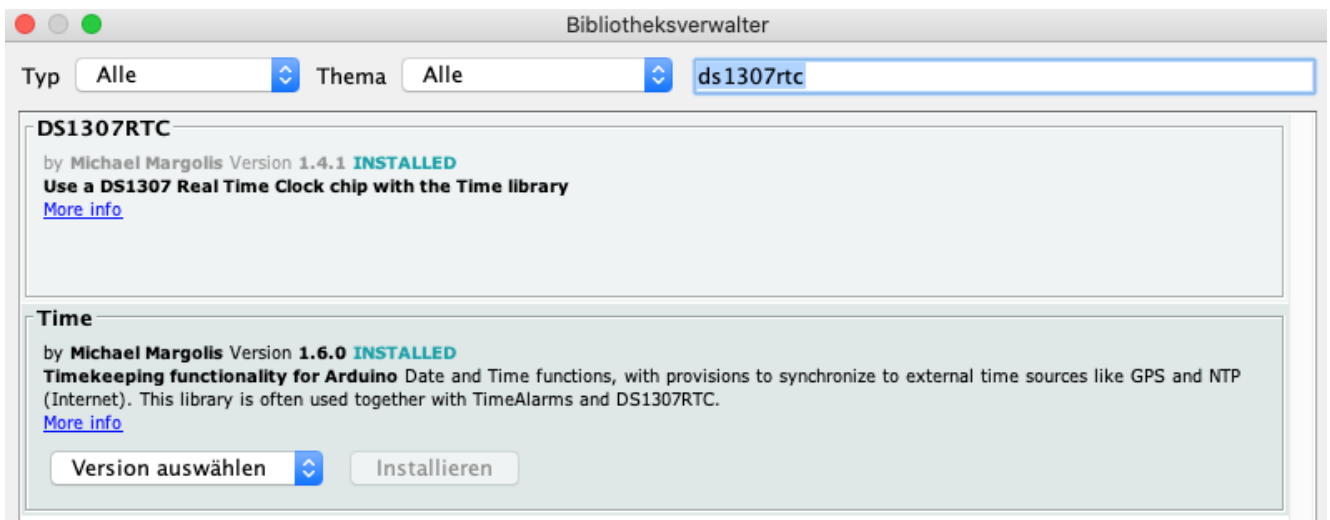
Dank der Verbindung per I²C benötigst du nur vier Kabel, um die Real Time Clock an deinem Arduino anzuschließen. Wenn du einen Arduino UNO verwendest, benötigst du (neben GND und 5V) hierfür die Pins A4 und A5. Orientiere dich an folgender Skizze:



Wenn du ein anderes Arduino-Modell verwendest, findest du die benötigten Pins in der [Arduino-Referenz](#). Am ESP8266 kommunizierst du mit den Pins D1 (SCL) und D2 (SDA) per I²C. Da der ESP8266 über WLAN verfügt, kannst du hier natürlich aber auch die [aktuelle Zeit von einem NTP-Server abfragen](#).

Die benötigten Bibliotheken

Um eine RTC mit dem Chip DS1307 verwenden zu können, benötigst du zwei Bibliotheken. Öffne hierfür den Bibliotheksverwalter und suche dort nach nach **DS1307rtc**. Du findest neben der gleichnamigen Bibliothek auch die Bibliothek **Time**. Installiere jeweils die aktuelle Version.



Sollte deine Real Time Clock einen anderen Chip haben, suche stattdessen nach dessen Namen. **Im Folgenden arbeiten wir jedoch mit dem DS1307 weiter.**

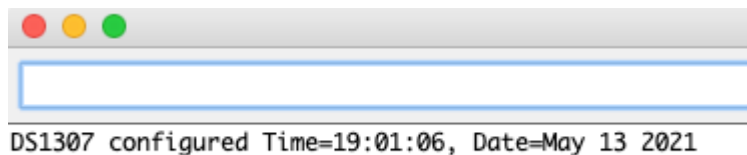
Wenn beide Bibliothek in deiner Arduino IDE installiert sind, kann es auch schon mit dem Einstellen der Uhr weitergehen.

Die aktuelle Uhrzeit in der RTC einstellen

Bevor in deinem RTC-Modul die Uhr ticken kann, muss es natürlich erst einmal wissen, wie viel Uhr es ist. Hierfür benötigst du einen Sketch, der bereits in den mitgelieferten Beispielen der Bibliothek **DS1307RTC** vorhanden ist.

Den Sketch findest du in der Arduino IDE unter Datei > Beispiele > DS1307RTC > **SetTime**

Öffne ihn und lade ihn auf deinen Arduino. Mit diesem Sketch „lädt“ dein Arduino die aktuelle Uhrzeit samt Datum von deinem Computer und speichert beides in der RTC.

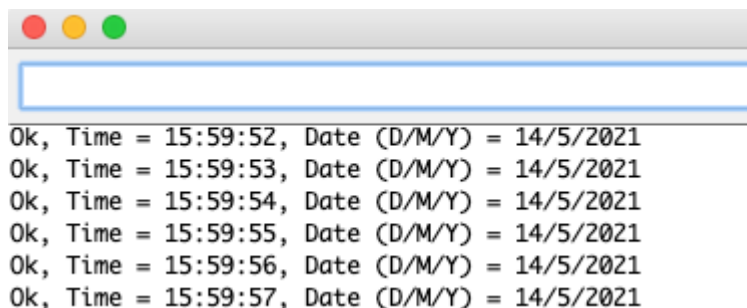


Das war es auch schon mit der Konfiguration. Den Sketch **SetTime** musst du erst wieder nach dem nächsten Batteriewechsel ausführen.

Uhrzeit und Datum im Seriellen Monitor ausgeben

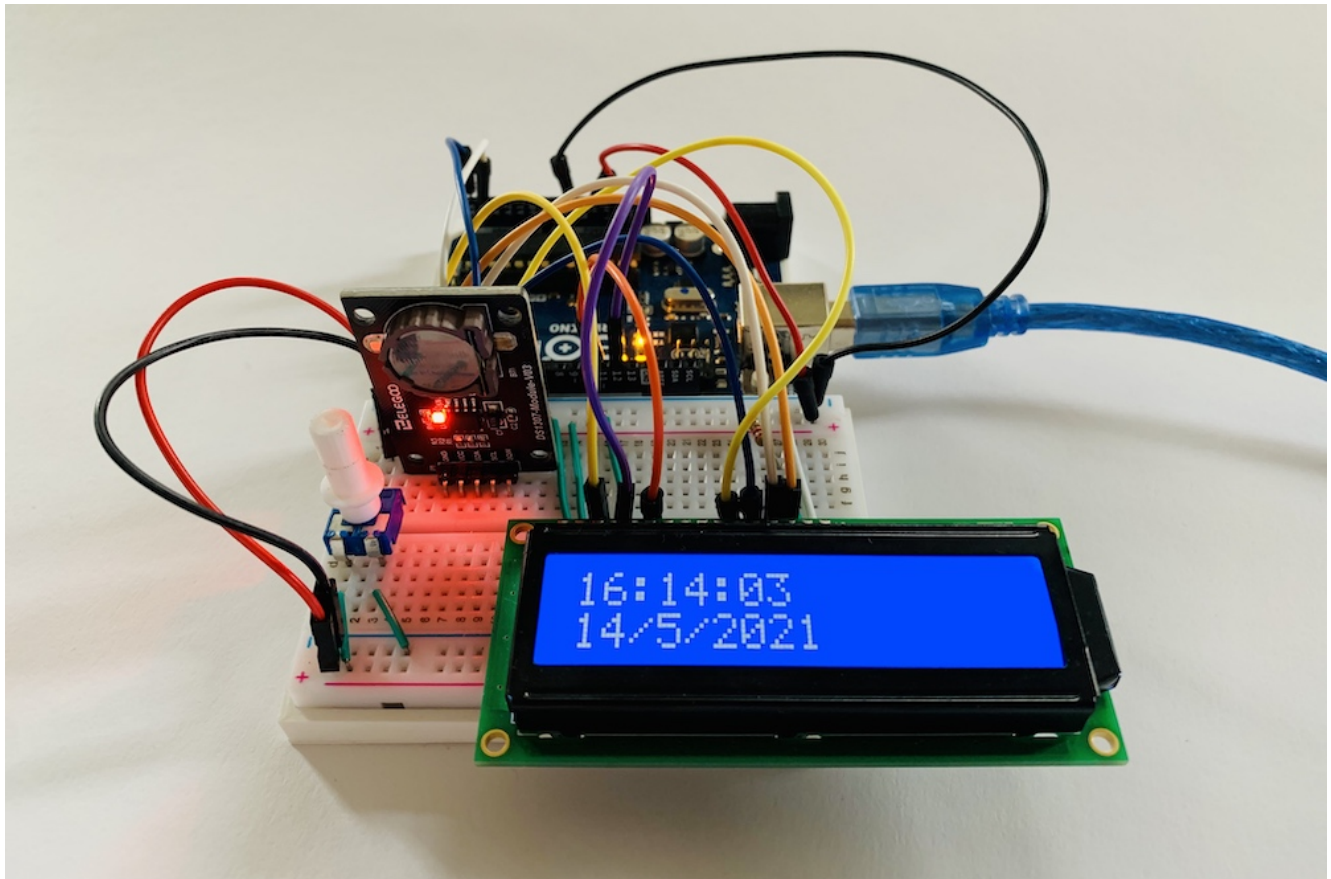
Für einen ersten Test reicht ebenfalls ein Beispiel-Sketch der Bibliothek. Öffne im gleichen Verzeichnis den Sketch **ReadTest** und lade ihn auf deinen Arduino.

In deinem Seriellen Monitor sollte nun im Sekundentakt die aktuelle Uhrzeit und das Datum erscheinen.



Uhrzeit und Datum auf einem LCD-Display anzeigen

Der Serielle Monitor ist das Eine – aber eine Digitaluhr ist etwas ganz Anderes. Schließe hierfür zunächst dein [LCD-Display am Arduino](#) zusätzlich zur RTC an.



Danach kommt der Sketch an die Reihe. Um es einfach zu halten, haben wir einfach den Sketch **ReadTest** modifiziert und um das Setup des LCD-Displays und der Ausgabe darauf erweitert.

Du findest im Loop nun unter jeder Ausgabe von Stunden, Minuten etc. auch den entsprechenden **lcd.print()** direkt darunter.

Ein besonderer Clou ist hierbei die Funktion **void print2digits()**, die auch schon im Sketch vorhanden ist. Diese Funktion sorgt dafür, dass jede Zahl zweistellig angezeigt

wird. Ohne diese Funktion würde die Uhrzeit 16:08 als 16:8 auf dem Display dargestellt.

Hier nun der gesamte Sketch:

```
#include <Wire.h>
#include <TimeLib.h>
#include <DS1307RTC.h>

#include <LiquidCrystal.h>
const int rs = 11, en = 12, d4 = 2, d5 = 3, d6 = 4, d7 = 5;
//Anschlüsse LCD -> Arduino
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {
  Serial.begin(9600);
  while (!Serial) ; // wait for serial
  delay(200);
  Serial.println("DS1307RTC Read Test");
  Serial.println("-----");
}

void loop() {

  // Festlegen, wie viele Zeichen und Reihen das Display hat:
  lcd.begin(16, 2);

  tmElements_t tm;

  if (RTC.read(tm)) {
    Serial.print("Ok, Time = ");
    print2digits(tm.Hour);
    lcd.setCursor(0, 0);
    lcd.print(tm.Hour);

    Serial.write(':');
    lcd.print(':');

    print2digits(tm.Minute);
    lcd.print(tm.Minute);

    Serial.write(':');
```

```

    lcd.print(':');

    print2digits(tm.Second);
    lcd.print(tm.Second);

    Serial.print(", Date (D/M/Y) = ");
    Serial.print(tm.Day);
    lcd.setCursor(0, 1);
    lcd.print(tm.Day);

    Serial.write('/');
    lcd.print('/');
    Serial.print(tm.Month);
    lcd.print(tm.Month);

    Serial.write('/');
    lcd.print('/');
    Serial.print(tmYearToCalendar(tm.Year));
    lcd.print(tmYearToCalendar(tm.Year));
    Serial.println();
} else {
    if (RTC.chipPresent()) {
        Serial.println("The DS1307 is stopped. Please run the
SetTime");
        Serial.println("example to initialize the time and begin
running.");
        Serial.println();
    } else {
        Serial.println("DS1307 read error! Please check the
circuitry.");
        Serial.println();
    }
    delay(9000);
}
delay(1000);
}

void print2digits(int number) {
    if (number >= 0 && number < 10) {
        Serial.write('0');
        lcd.print('0');
    }
}

```

```
}  
Serial.print(number);  
}
```

Wie geht es weiter?

Weitere Anleitungen zu Bauteilen und vollständige Arduino Projekte findest du in unserem Buch für angehende Maker:



[Vom Anfänger zum Maker: Mit dem Arduino und ESP8266](#)

▪ Kumbartzki, Frederik (Autor)

[Bei Amazon kaufen](#)

Du hast nun die Möglichkeit, in deinem Projekt immer die aktuelle Uhrzeit zu verwenden. Welche Ideen kommen dir nun in den Sinn? Wie wäre es z.B. mit einem Wecker, der dich mit

einem [Piezo](#) weckt?