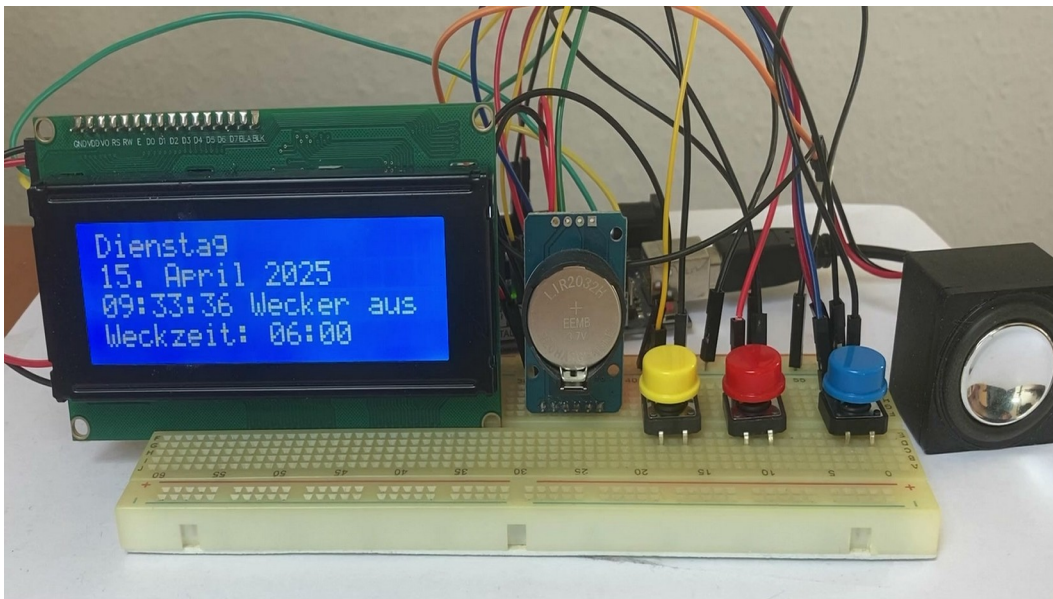


Ziel des Projekts

Die Weckzeit wird zusammen mit Wochentag, Datum und Uhrzeit auf dem LCD-Display angezeigt.

Die Funktionen der Taster:

- ◆ links: Stunden vorwärts
- ◆ mittlerer: Minuten vorwärts
- ◆ rechter: Wecker ein-/ausschalten

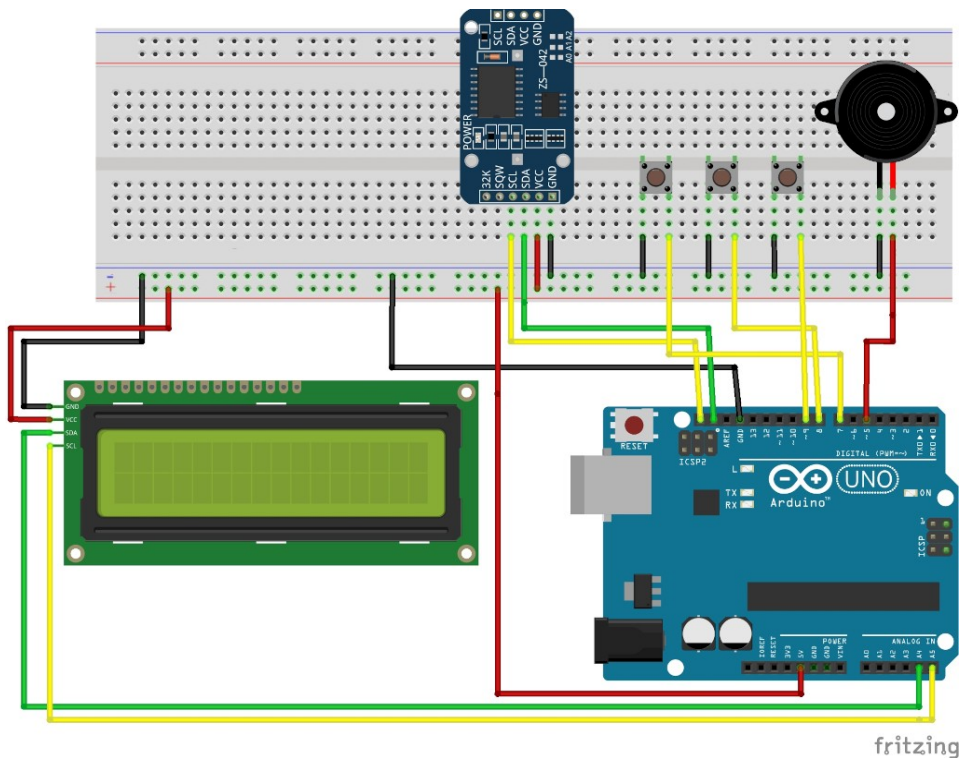


Benötigte Bauteile

- ➔ 3 Taster
- ➔ 4-zeiliges LCD mit I²C-Schnittstelle
- ➔ RTC-Modul DS3231
- ➔ Lautsprecher
- ➔ Leitungsdrähte

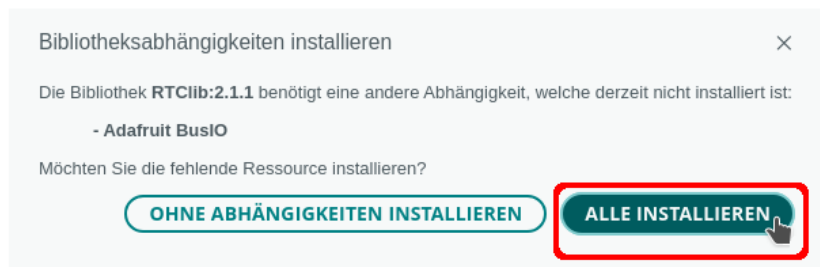
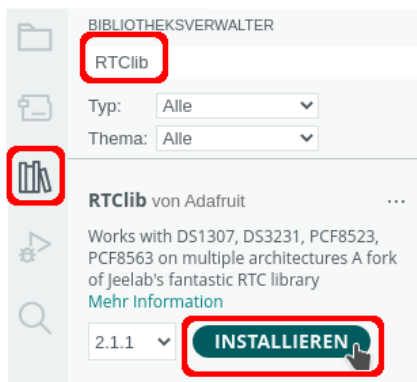
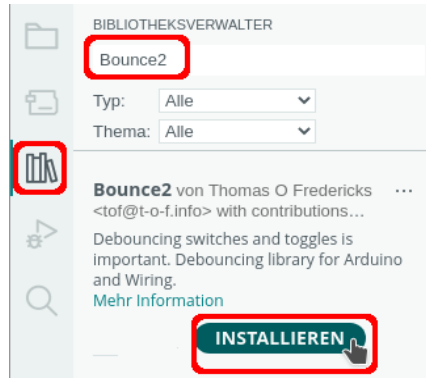
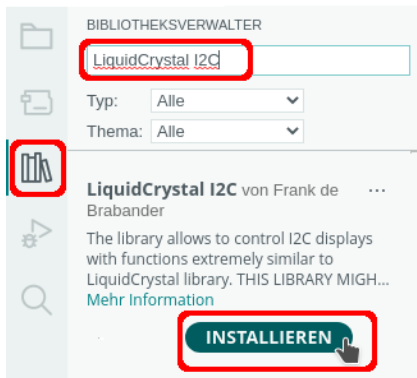


Der Schaltplan



Benötigte Bibliotheken

Sketch → Bibliothek einbinden → Bibliotheken verwalten





Das Programm

Binde die benötigten Bibliotheken ein und definiere die Variablen:

```
#include "RTCLib.h"
#include "LiquidCrystal_I2C.h"
#include "Bounce2.h"

// Name des RTC-Moduls
RTC_DS3231 rtc;

// 4-zeiliges LCD
LiquidCrystal_I2C lcd(0x27, 20, 4);

int Lautsprecher = 3;
int TasterLinks = 7;
int TasterMitte = 8;
int TasterRechts = 9 ;

// Beginn der Weckzeit
int StundeWeckzeit = 6;
int MinuteWeckzeit = 0;

// Dauer des Signal in Sekunden
int DauerWecksignal = 3;
bool TonSekunde = false;
bool WeckerEinAus = true;

// Zeit messen für Ticken der Uhr
long Startzeit;
long ZeitJetzt;
Bounce WeckzeitStunde = Bounce();
Bounce WeckzeitMinute = Bounce();
Bounce StatusWecker = Bounce();
```

Der setup-Teil. Beachte die Kommentare.

```
void setup()
{
  pinMode(TasterLinks, INPUT_PULLUP);
  pinMode(TasterMitte, INPUT_PULLUP);
  pinMode(TasterRechts, INPUT_PULLUP);

  // Instanzen des Objekts Bounce für jede Taste zuordnen
  // Zeitintervall einstellen
  WeckzeitStunde.attach(TasterLinks);
  WeckzeitStunde.interval(20);
  WeckzeitMinute.attach(TasterMitte);
  WeckzeitMinute.interval(20);
  StatusWecker.attach(TasterRechts);
  StatusWecker.interval(20);

  Serial.begin(9600);
```



```
// auf serielle Verbindung warten
while (!Serial) {}

rtc.begin();
/*
  wenn Datum und Zeit nicht korrekt -> Datum/Zeit setzen
  Jahr, Monat, Tag, Stunde, Minute, Sekunde
  keine führende 0 setzen
  rtc.adjust(DateTime(2025, 4, 14, 16, 42, 30));
*/
// rtc.adjust(DateTime(2025, 4, 14, 16, 42, 30));

// Zeitpunkt des Kompilierens als aktuelle Zeit setzen
rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));

// LCD starten
lcd.init();

// Hintergrundbeleuchtung einschalten
lcd.backlight();

Serial.println("Bitte die Weckzeit eingeben:");
Serial.println("Taster links -> Stunde +/Taster Mitte -> Minute +");
Serial.println("Taster rechts -> Weckzeit ein/aus");
Startzeit = millis();
}
```

Der loop-Teil:

```
void loop()
{
  String Stunde, Minute, Weckzeit;

  // aktuelle Zeit lesen
  DateTime aktuell = rtc.now();
  // Stunde einstellen
  if (WeckzeitStunde.update())
  {
    if (WeckzeitStunde.read() == LOW)
    {
      if (StundeWeckzeit >= 24) StundeWeckzeit = 1;
      else StundeWeckzeit ++;
    }
  }

  // Minute einstellen
  if (WeckzeitMinute.update())
  {
    if (WeckzeitMinute.read() == LOW)
    {
      if (MinuteWeckzeit >= 59) MinuteWeckzeit = 0;
      else MinuteWeckzeit ++;
    }
  }
}
```



```
// Wecker ein/aus
if (StatusWecker.update())
{
    if (StatusWecker.read() == LOW)
    {
        WeckerEinAus = !WeckerEinAus;
    }
}

// Wochentag, Datum und Zeit anzeigen
lcd.setCursor(0, 0);

/*
    Wochentag anzeigen
    0 = Sonntag
    1 = Montag
    ...
    6 = Samstag
*/
switch (aktuell.dayOfTheWeek())
{
    case 0:
        lcd.print("Sonntag");
        break;
    case 1:
        lcd.print("Montag");
        break;
    case 2:
        lcd.print("Dienstag");
        break;
    case 3:
        lcd.print("Mittwoch");
        break;
    case 4:
        lcd.print("Donnerstag");
        break;
    case 5:
        lcd.print("Freitag");
        break;
    case 6:
        lcd.print("Samstag");
        break;
}

lcd.setCursor(0, 1);
lcd.print(aktuell.day());
lcd.print(".");

// Monatsnamen anzeigen
lcd.print(" ");

switch (aktuell.month())
{
    case 1:
        lcd.print("Januar");
        break;
```



```
case 2:
    lcd.print("Februar");
    break;
case 3:
    lcd.print("M\341rz");
    break;
case 4:
    lcd.print("April");
    break;
case 5:
    lcd.print("Mai");
    break;
case 6:
    lcd.print("Juni");
    break;
case 7:
    lcd.print("Juli");
    break;
case 8:
    lcd.print("August");
    break;
case 9:
    lcd.print("September");
    break;
case 10:
    lcd.print("Oktober");
    break;
case 11:
    lcd.print("November");
    break;
case 12:
    lcd.print("Dezember");
    break;
}
lcd.print(" ");

lcd.print(aktuell.year());
lcd.setCursor(0, 2);

// wenn Stunden < 10 -> führende 0 setzen
if (aktuell.hour() < 10) lcd.print("0");
lcd.print(aktuell.hour());
lcd.print(':');

// wenn Minuten < 10 -> führende 0 setzen
if (aktuell.minute() < 10) lcd.print("0");
lcd.print(aktuell.minute());
lcd.print(':');

// wenn Sekunden < 10 -> führende 0 setzen
if (aktuell.second() < 10) lcd.print("0");
lcd.print(aktuell.second());

if (WeckerEinAus) lcd.print(" Wecker ein");
else lcd.print(" Wecker aus");
```



```
/*
    Weckzeit formatieren -> führende 0 ergänzen
    4 mögliche Fälle
    Stunde < 10 Minute < 10
    Stunde < 10 Minute > 9
    Stunde > 10 Minute < 10
    Stunde > 10 Minute > 9
*/
if (StundeWeckzeit < 10 && MinuteWeckzeit < 10)
{
    Weckzeit = "0" + String(StundeWeckzeit) + ":0" + String(MinuteWeckzeit);
}

if (StundeWeckzeit < 10 && MinuteWeckzeit > 9)
{
    Weckzeit = "0" + String(StundeWeckzeit) + ":" + String(MinuteWeckzeit);
}

if (StundeWeckzeit > 9 && MinuteWeckzeit < 10)
{
    Weckzeit = String(StundeWeckzeit) + ":0" + String(MinuteWeckzeit);
}

if (StundeWeckzeit > 9 && MinuteWeckzeit > 9)
{
    Weckzeit = String(StundeWeckzeit) + ":" + String(MinuteWeckzeit);
}
lcd.setCursor(0, 3);
lcd.print("Weckzeit: " + Weckzeit);

// führende 0 setzen
if (aktuell.hour() < 10) Stunde = "0" + String(aktuell.hour());
else Stunde = String(aktuell.hour());

// führende 0 setzen
if (aktuell.minute() < 10) Minute = "0" + String(aktuell.minute());
else Minute = String(aktuell.minute());

// String zusammensetzen
String aktuelleZeit = Stunde + ":" + Minute;
if (aktuelleZeit == Weckzeit && aktuell.second() < DauerWecksignal)
{
    tone(Lautsprecher, 1000, 500);
}

/*
    Ticken der Uhr
    aktuelle Zeit (ZeitJetzt) messen
    millis() -> Zeit, die seit dem Start des Programms
    vergangen ist
    wenn die Differenz zwischen der gerade gemessenen Zeit
    und der im setup gemessenen Startzeit >= 1000 (1 Sekunde) ist
    -> kurzen Ton wiedergeben
    Startzeit muss anschließend auf die aktuelle Zeit gesetzt werden
*/
```



```
ZeitJetzt = millis();  
if (ZeitJetzt - Startzeit >= 1000)  
{  
  if (TonSekunde) tone(Lautsprecher, 1000, 2);  
  Startzeit = ZeitJetzt;  
}
```

Hartmut Waller letzte Änderung: 24.11.25