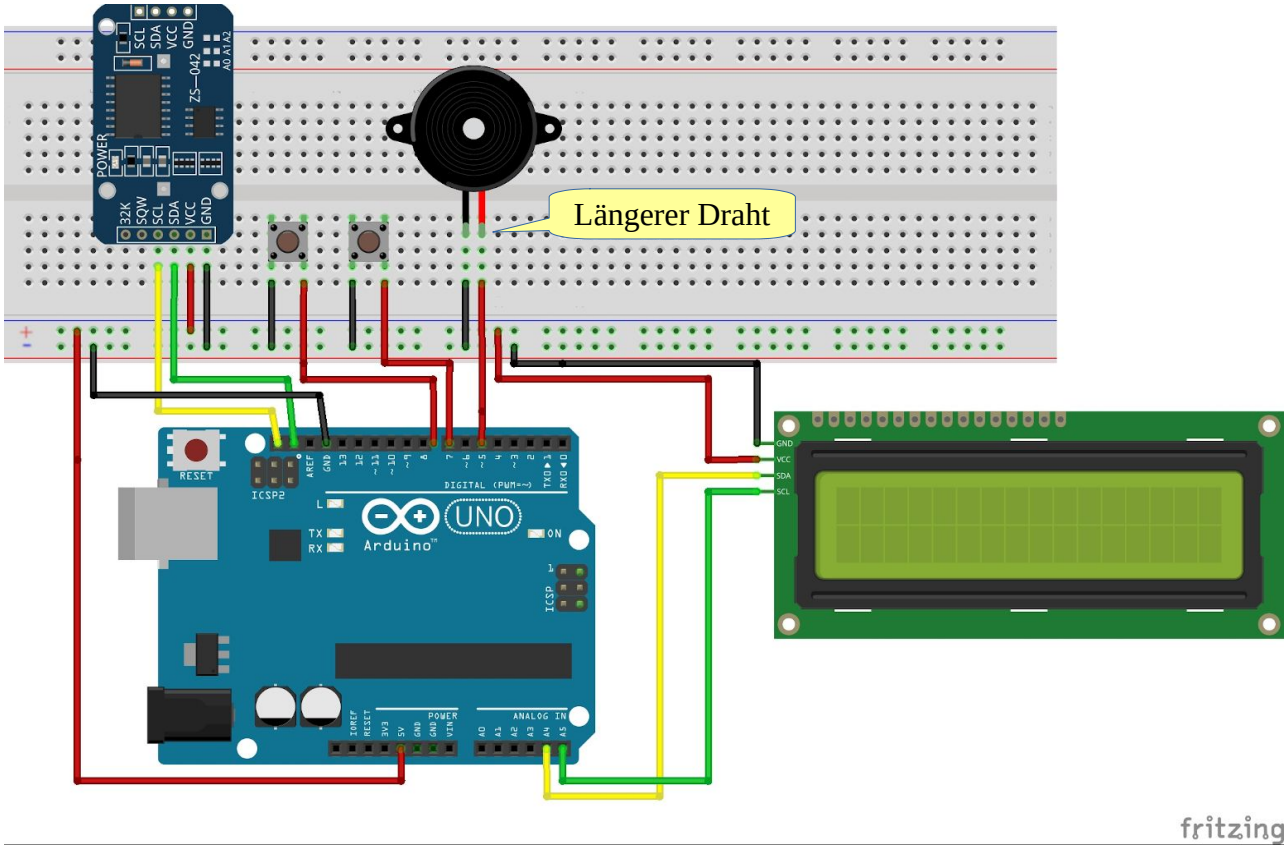
Wecker mit einem RTC-Modul

Benötigte Bauteile:

- 2 Taster
- LCD 1602
- RTC-Modul DS3231
- Lautsprecher
- Leitungsdrähte



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

Der linke Taster soll die Stunden, der rechte Taster soll die Minuten einstellen.
Ein Tasterdruck → eine Stunde/eine Minute vorwärts

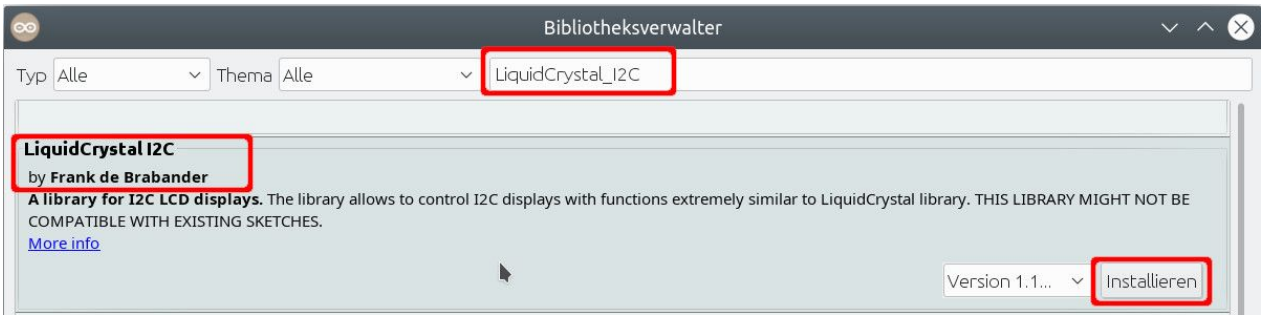
Beginne mit der Definition der Bauteile und der voreingestellten Weckzeit:

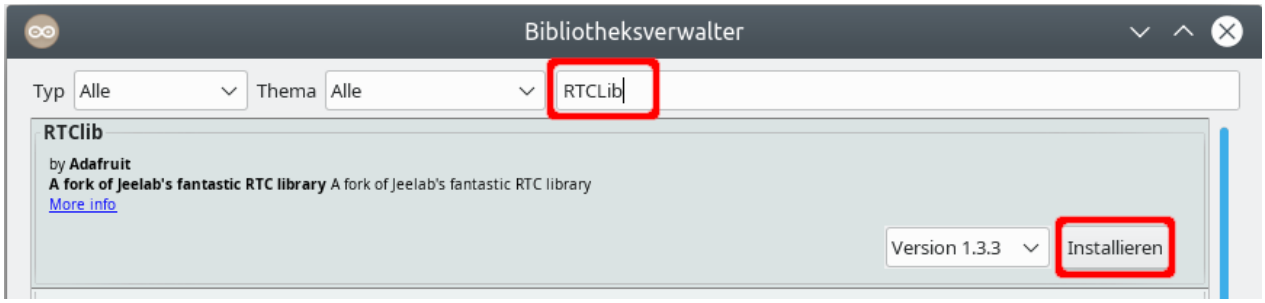
```
int LAUTSPRECHER = 5;
int TASTERLINKS = 7;
int TASTERRECHTS = 8;
int TasterLesenLinks;
int TasterLesenRechts;
```

```
// Beginn der Weckzeit
int StundeWeckzeit = 6;
int MinuteWeckzeit = 0;
int DauerWecksignal = 3;

// Zeit messen für Ticken der Uhr
long Startzeit;
long ZeitJetzt;
```

Benötigte Bibliotheken
Sketch → Bibliothek einbinden → Bibliotheken verwalten





Binde die benötigten Bibliotheken ein und definiere die Variablen:

```
# include <RTClib.h>
# include <LiquidCrystal_I2C.h>

// Name des RTC-Moduls
RTC_DS3231 rtc;

// Name des LCD-Moduls
LiquidCrystal_I2C lcd(0x27, 20, 4);

int LAUTSPRECHER = 3;
int TASTERLINKS = 7;
int TASTERRECHTS = 8;
int TasterLesenLinks;
int TasterLesenRechts;
```

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

// Dauer des Signal
int DauerWecksignal = 3;

// Zeit messen für Ticken der Uhr
long Startzeit;
long ZeitJetzt;
```

Der setup-Teil. Beachte die Kommentare.

```
void setup ()
{
  pinMode(TASTERLINKS, INPUT_PULLUP);
  pinMode(TASTERRECHTS, INPUT_PULLUP);

  Serial.begin(9600);
  rtc.begin();
  /*
    wenn Datum und Zeit nicht korrekt -> Datum/Zeit setzen
    Jahr, Monat, Tag, Stunde, Minute, Sekunde
    keine führende 0 setzen
    Beispiel:
    rtc.adjust(DateTime(2018, 10, 25, 7, 2, 30));
  */
  // rtc.adjust(DateTime(2020, 10, 28, 13, 26, 30));
  lcd.init();
  lcd.backlight();
  Serial.println("Bitte die Weckzeit eingeben:");
  Serial.println("Taster links -> Stunde +/Taster rechts -> Minute +");
  Startzeit = millis();
}
```

Der loop-Teil. Beachte die Kommentare.

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

  // aktuelle Zeit lesen
  DateTime aktuell = rtc.now();

  // Stunde einstellen
  TasterLesenLlinks = digitalRead(TASTERLINKS);

  // Minute einstellen
  if (TasterLesenLinks == LOW)
  {
    delay(200);
```

```
  if (StundeWeckzeit >= 24) StundeWeckzeit = 1;
  else StundeWeckzeit ++;
}

TasterLesenRechts = digitalRead(TASTERRECHTS);
if (TasterLesenRechts == LOW)
{
  delay(200);
  if (MinuteWeckzeit >= 59) MinuteWeckzeit = 0;
  else MinuteWeckzeit ++;
}

// 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 0:
    lcd.print("Januar");
    break;
  case 1:
    lcd.print("Februar");
    break;
  case 2:
    lcd.print("M\341rz");
    break;
  case 3:
    lcd.print("April");
    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());

/*
  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)
  {
    tone(LAUTSPRECHER, 1000, 2);
    Startzeit = ZeitJetzt;
  }
}
```