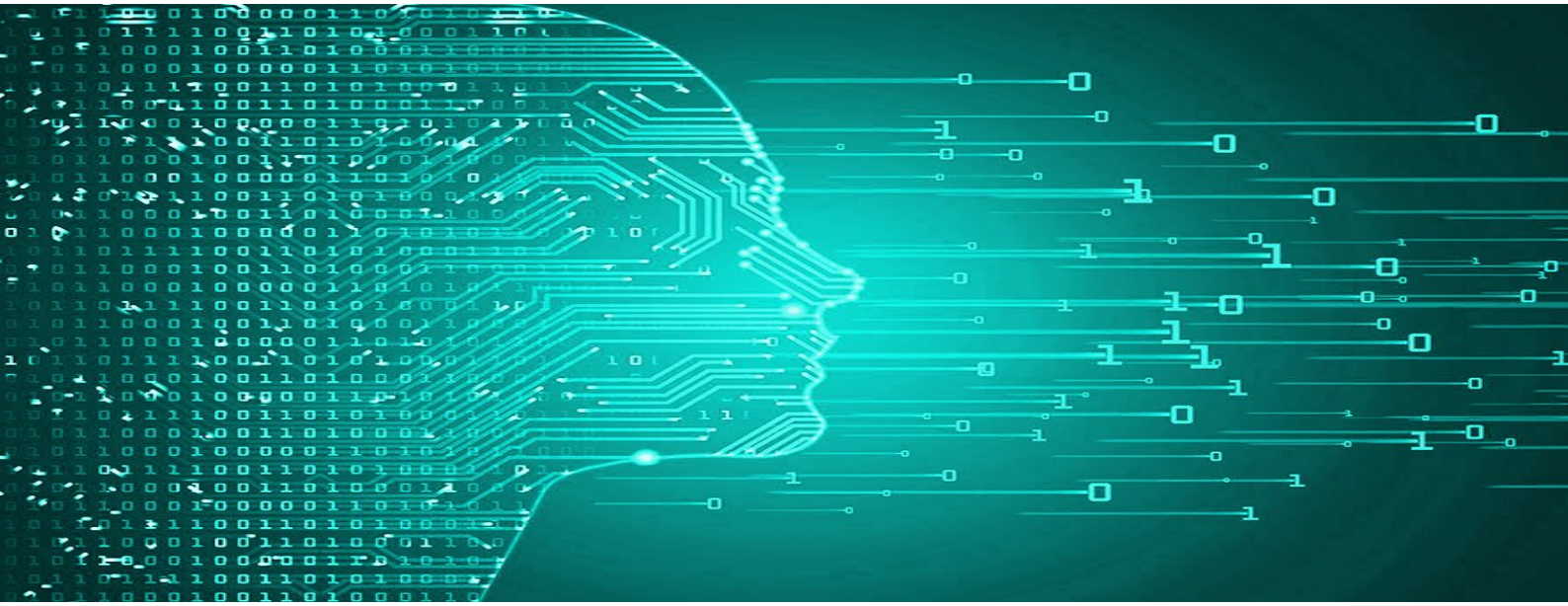


Elmar Wings

AI with an Arduino Nano 33 BLE Sense

Realtime Object Detection with the ArduCAM

Version: 1765
May 14, 2025

Contents

Contents 3

List of Figures 7

List of Listings 9

1. Einleitung 13
1.1. Herausforderungen . . . 13
1.2. Lösungen . . . 13
1.3. Struktur des Dokuments . . . 13

I. Arduino Nano 33 BLE Sense 15

2. Arduino IDE 2.3.6 17
2.1. Beschreibung der Arduino IDE . . . 17
2.2. Installation der Arduino IDE . . . 18
2.2.1. Download der Arduino IDE . . . 18
2.2.2. Installation auf Windows . . . 18
2.2.3. Installation (MacOS) . . . 23
2.3. Übersicht . . . 23
2.4. Funktionen . . . 24
2.4.1. Sketchbook . . . 24
2.4.2. Board-Verwaltung . . . 24
2.4.3. Bibliotheksverwalter . . . 25
2.5. Integration und Verwendung einer benutzerdefinierten Bibliothek in der Arduino IDE . . . 26
2.5.1. Erstellung der Bibliotheksdateien . . . 26
2.5.2. Installation und Integration der Bibliothek . . . 27
2.5.3. Verwendung symbolischer Links zur Bibliotheksintegration . . . 27
2.5.4. Windows-Tool `mklink` zur Erstellung symbolischer Links . . . 28
2.5.5. Überprüfung der Bibliotheksverfügbarkeit in der Arduino IDE 29
2.5.6. Serieller Monitor . . . 29
2.5.7. Serieller Plotter . . . 30
2.6. Beispiel-Sketches . . . 30
2.7. Debugging . . . 30
2.8. Autovervollständigung . . . 31
2.9. Zusammenfassung . . . 31

3. Einrichtung des Arduino Nano 33 BLE Sense 33
3.1. Einführung . . . 33
3.2. Konfiguration des Arduino Nano 33 BLE Sense . . . 33
3.2.1. Verbindung und Konfiguration des Arduino-Boards mit dem Computer . . . 34
3.2.2. Auswahl des richtigen Ports . . . 35
3.2.3. Sketch verifizieren und hochladen . . . 37

3.3. Konfiguration testen . . . 38
3.3.1. Test der Einrichtung mit dem Beispiel-Sketch `blink.ino` . . . 38

4. Arduino Nano 33 BLE Sense 41
4.1. Arduino Nano 33 BLE Sense . . . 41
4.2. Unterschied zwischen dem Arduino Nano 33 BLE Sense Rev1, dem Arduino Nano 33 BLE Sense Rev2 und dem Arduino Nano 33 BLE Sense Lite . . . 43
4.2.1. Worin unterscheiden sich Rev1 und Rev2? . . . 43
4.2.2. Arduino Nano 33 BLE Sense Lite . . . 44
4.3. Beschreibung der On-Board Sensoren . . . 45
4.4. Pinbelegung des Arduino Nano 33 BLE . . . 47

II. Harwarebeschreibung: Magic Faucet Fountain 49

5. Materialliste Magic Faucet Fountain 51

6. Ultraschallsensor HC-SR04 53
6.1. Einleitung . . . 53
6.2. Ultraschallsensoren zur Abstandsmessung . . . 53
6.3. Grundlagen zur Verwendung eines Ultraschallsensors . . . 54
6.3.1. Grundlagen Schall . . . 54
6.3.2. Schallgeschwindigkeit und Wellenlänge . . . 54
6.3.3. Weg- und Abstandsmessung mit Ultraschall . . . 55
6.4. Ultraschall-Abstandsensor HC-SR04 . . . 56
6.5. Spezifikation . . . 57
6.5.1. Anwendungen . . . 57
6.6. Bibliothek . . . 57
6.6.1. Beschreibung . . . 57
6.6.2. Installation . . . 57
6.6.3. Funktionen . . . 58
6.7. Kalibrierung . . . 58
6.8. Simple Code . . . 58
6.9. Einfache Anwendung . . . 58
6.10. Tests . . . 62
6.10.1. Einfacher Funktionstest . . . 62
6.10.2. Test aller Funktionen . . . 63
6.10.3. Lösungsansätze . . . 64
6.11. Weiterführende Anwendungen . . . 64
6.12. Further Readings . . . 65

7. Comet Tauchpumpe ELEGANT 12V 67
7.1. Einführung . . . 67
7.1.1. Arbeitsweise Kreiselpumpen . . . 67
7.2. Charakteristische Größen von Pumpen . . . 68
7.2.1. Volumenstrom . . . 68
7.2.2. Förderhöhe . . . 68
7.2.3. Nutzleistung . . . 68
7.2.4. Pumpenwirkungsgrad . . . 69
7.2.5. Kupplungsleistung . . . 69
7.2.6. Anlagenkennlinie . . . 69
7.3. Comet Tauchpumpe ELEGANT 12V . . . 69
7.4. Spezifikation . . . 70
7.4.1. Technische Daten: . . . 70

7.4.2. Weitere Eigenschaften . . . 70
7.5. Einfacher Code . . . 70
7.6. Tests . . . 72
7.7. Einfache Anwendung . . . 72
7.8. Further Readings . . . 74

8. Schaltnetzteil: MW RD-50A 75
8.1. Einleitung . . . 75
8.2. Grundlegende Informationen . . . 75
8.3. Schaltnetzteil: MW RD-50A . . . 76
8.4. Spezifikation . . . 76
8.5. Tests . . . 77
8.6. Einfache Anwendung . . . 77
8.7. Further Readings . . . 80

9. Bluetooth Modul HC-05 81
9.1. Einleitung . . . 81
9.2. Grundlegende Informationen . . . 81
9.2.1. Bluetooth Varianten . . . 81
9.2.2. Reichweite und Sendeleistung . . . 82
9.2.3. Daten Übertragung . . . 82
9.2.4. Anwendungen . . . 82
9.3. Bluetooth Modul: HC-05 . . . 83
9.4. Spezifikation . . . 83
9.4.1. Hardware-Spezifikation . . . 83
9.4.2. Elektrische Eigenschaften . . . 84
9.4.3. Schnittstellen und Anschlüsse . . . 84
9.4.4. Mechanische und Umgebungsbedingungen . . . 85
9.4.5. Zusätzliche Hardware-Features . . . 86
9.4.6. Pinbelegung (Auszug) . . . 86
9.5. Bibliotheken . . . 86
9.6. Einfacher Code . . . 87
9.7. Tests . . . 88
9.8. Einfache Anwendung . . . 88
9.9. Further Readings . . . 90

10. Kippschalter: GOOBAY 10013 91
10.1. Einleitung . . . 91
10.2. Grundlegende Informationen . . . 91
10.3. Schalter: GOOBAY 10013 . . . 91
10.4. Spezifikation . . . 91
10.5. Tests . . . 92
10.6. Einfache Anwendung . . . 92

11. MOS-FET 95
11.1. Einleitung . . . 95
11.2. Grundlegende Informationen . . . 95
11.2.1. N-Kanal-MOS-FET . . . 96
11.2.2. P-Kanal-MOS-FET . . . 96
11.3. MOS-FET IRF9540N . . . 96
11.4. Spezifikation . . . 96
11.5. Einfacher Code . . . 98
11.6. Tests . . . 99
11.7. Einfache Anwendung . . . 99

12. Externe LED **101**
12.1. Einleitung . . . 101
12.2. Grundlegende Informationen . . . 101
12.3. LED CML Signalleuchte, blau, ultrahell, 12 VDC . . . 102
12.4. Spezifikation . . . 102
12.5. Einfacher Code . . . 102
12.6. Tests . . . 103
12.7. Einfache Anwendung . . . 103
12.8. Further Readings . . . 104

III. Applikation: Magic Faucet Fountain **105**

13. Projekt: Magic Faucet Fountain **107**
13.1. Beschreibung . . . 107
13.2. Aufbau . . . 107
13.2.1. Abmaß . . . 108
13.3. Hardware-Schnittstellen . . . 108
13.3.1. Schaltpläne . . . 108
13.4. Software-Schnittstellen . . . 111
13.4.1. Ablaufdiagramm der Anwendung . . . 112
13.4.2. Code der Anwendung . . . 114
13.5. Installation . . . 115
13.5.1. Mechanischer Aufbau . . . 115
13.5.2. Elektrische Installation . . . 115
13.5.3. Software und App-Verbindung . . . 116
13.5.4. Software und App-Verbindung . . . 116
13.6. Konfiguration . . . 116
13.7. Einschränkungen . . . 117

14. Schlussfolgerungen **119**
14.1. Zusammenfassung der Projektergebnisse . . . 119
14.2. Reflexion über die Zielerreichung . . . 119
14.3. Mögliche Weiterentwicklungen . . . 119
14.4. Zukünftige Anwendungen . . . 119

IV. Anhang **121**

Literaturverzeichnis **123**

Stichwortverzeichnis **126**

Index **127**

List of Figures

2.1. Menü-Leiste . . . 17
2.2. Arduino IDE 2.3.6 . . . 18
2.3. Option für Windows . . . 19
2.4. Lizenzabkommen . . . 20
2.5. Auswahl der Benutzer . . . 21
2.6. Auswahl des Installationsverzeichnis . . . 22
2.7. Die Arduino IDE . . . 22
2.8. Die neue Arduino IDE mit der nützlichen Sidebar . . . 23
2.9. Das Sketchbook auf der Arduino IDE . . . 24
2.10. Die Board-Verwaltung auf der Arduino IDE . . . 25
2.11. Der Bibliotheksverwalter auf der Arduino IDE . . . 26
2.12. Administrator-Eingabeaufforderung in Windows . . . 28
2.13. Einbindung der Nano33BLESenseLED-Bibliothek . . . 29
2.14. Beispielsketche in der Arduino IDE . . . 30
2.15. Autovervollständigen in der Arduino-IDE . . . 31

3.1. Mbed OS Nano-Paket installation . . . 34
3.2. Verbinden des Mbed OS Nano-Pakets mit dem Board . . . 34
3.3. Arduino Nano 33 BLE Sense Reset Button . . . 35
3.4. green and orange Builtin-LED . . . 36
3.5. Selektrieren des verfügbaren Ports . . . 37
3.6. Navigation zum Blink-Beispiel in der Arduino IDE . . . 38
3.7. Code des Blink-Beispiels in der Arduino IDE . . . 39

4.1. Arduino Nano 33 BLE Sense Rev. 2, see Arduino Store . . . 42
4.2. Arduino Nano 33 BLE Sense Rev. 1 . . . 44
4.3. Arduino Nano 33 BLE Sense Lite . . . 45
4.4. . . . 46
4.5. Pinbelegung des Arduino Nano 33 BLE Sense . . . 47

6.1. Frequenz der verschiedenen Schall-Arten . . . 54
6.2. Ausbreitungsgeschwindigkeit von Schall in Luft . . . 55
6.3. Spannungsverlauf Schallwandler bei einer Echo-Laufzeit-Messung, nach [HS23] . . . 56
6.4. Ultraschallsensor mit Pin-Belegung . . . 56

8.1. Aufbau eines Schaltnetzteils . . . 76
8.2. Skizze des Netzteils . . . 77
8.3. Spannungsmessung des Netzteils . . . 77
8.4. Anwendung des Netzteils für 2 seperate Stromkreise . . . 78

9.1. Skizze des Bluetooth Moduls HC-05 . . . 83
9.2. Schnittstellen und Anschlüsse HC-05 . . . 84

10.1. Skizze der Schaltertypen . . . 91
10.2. Messen des Widerstands . . . 92
10.3. Einfache Anwendung des Schalters . . . 93

11.1. Skizze des MOS-FETs 96
11.2. Pinbelegung des MOS-FETs 98

12.1. Aufbau einer LED 102

13.1. Skizze des Magic Faucet Fountain 107
13.2. Versorgungsschaltplan 109
13.3. Schaltplan: Arduino-basierte Steuerungseinheit mit Bluetooth-Kommunikation 110
13.4. 5/12 V Netz mit Aktoren 111
13.5. Ablaufdiagramm 112

List of Listings

6.1. Einfacher Code zum Testen des Ultraschallsensors 59
6.2. Einfache Anwendung als Schwellenwert-Melder 61
6.3. Code für die Messung einer Strecke 63

7.1. Einfacher Code zur Ansteuerung einer Pumpe 71
7.2. Einfacher Code zur Fehlererkennung 73

8.1. Einfacher Code zum Verwenden des Netzteils 79

9.1. Code zur Überprüfung der Funktionalität 88
9.2. Code zur Überprüfung der Funktionalität 89

11.1. Einfacher Code zum Testen des MOS-FETs 99
11.2. Einfache Anwendung zum Verwenden des MOS-FETs 100

12.1. Einfacher Code zur Ansteuerung einer externen LED 103
12.2. Einfacher Code zum Blinken einer externen LED 104

Acronyms

GND Ground

GPIO General Purpose Input Output

I^{2}C Inter-Integrated Circuit

IMU Inertial Measurement Unit

SCL Serial Clock

SDA Serial Data

UART Universal Asynchronous Receiver Transmitter

VCC Voltage at the Common Collector

1. Einleitung

1.1. Herausforderungen

1.2. Lösungen

1.3. Struktur des Dokuments

Part I.

Arduino Nano 33 BLE Sense

2. Arduino IDE 2.3.6

Dieses Kapitel bietet eine umfassende Untersuchung der Arduino IDE, einschließlich ihrer Funktionen, Schnittstellenoptionen und Funktionalität. Es enthält eine detaillierte Erklärung des Zwecks und der Nutzbarkeit jeder Komponente, eine schrittweise Installationsanleitung sowie Anweisungen zur Initialisierung und Konfiguration der Umgebung. Dieser grundlegende Überblick befähigt die Leser, die IDE effektiv für die Programmierung und Fehlerbehebung von Arduino-kompatibler Hardware zu nutzen.

2.1. Beschreibung der Arduino IDE

Die Arduino IDE ist eine Open-Source-Software, die speziell für das Bearbeiten, Kompilieren und Hochladen von Code auf Arduino-Module entwickelt wurde. Sie ist plattformübergreifend und mit Betriebssystemen wie Windows, Linux und macOS kompatibel. Auf der Java-Plattform aufbauend unterstützt die IDE verschiedene Arduino-Module und die Programmierung in C und C++.
Die Mikrocontroller auf den Arduino-Boards werden so programmiert, dass sie die in Form von Code bereitgestellten Informationen verarbeiten. Programme, die in der IDE geschrieben werden, werden als Skizzen (Sketches) bezeichnet. Diese Skizzen generieren eine Hex-Datei, die dann auf den Mikrocontroller übertragen und hochgeladen wird.
Die IDE-Umgebung besteht aus zwei Hauptkomponenten: dem Editor und dem Compiler. Der Editor wird zum Schreiben von Code verwendet, während der Compiler den Code kompiliert und auf das Arduino-Modul hochlädt.[FAD18] In Version 2.3.3 der Arduino IDE spielt die Menüleiste, die sich am oberen Rand der Schnittstelle befindet, eine entscheidende Rolle, da sie Zugriff auf wichtige Befehle wie Hochladen, Verifizieren/Kompilieren und Speichern bietet. Zusätzlich enthält sie das Dateimenü zum Öffnen neuer oder bestehender Dateien sowie den Beispiele-Bereich, der vorgefertigte Skizzen für verschiedene Anwendungen wie Blink und Fade bereitstellt.

Die 6 Buttons am oberen Rand des Fensters sind wie folgt in 2.1:

Figure 2.1.: Menü-Leiste

- Das Häkchen-Symbol dient zum Überprüfen des geschriebenen Codes. Nach der Code-Eingabe prüft diese Funktion auf Fehler und bestätigt die korrekte Struktur. Dieser Schritt stellt sicher, dass der Code für die Hardware bereit ist.

- Das Hochladen-Symbol kompiliert den Code und überträgt ihn auf das angeschlossene Board, sodass dieser ausgeführt werden kann.

- Das Debug-Symbol startet eine Debugging-Session, mit der Sie den Code analysieren können – inklusive Breakpoints, Schritt-für-Schritt-Ausführung und Variableninspektion (auf kompatiblen Boards).

Das Serieller Plotter-Symbol öffnet ein Tool zur Visualisierung von Echtzeitdaten, die über die serielle Verbindung vom Board gesendet werden, und stellt sie als Graphen dar.

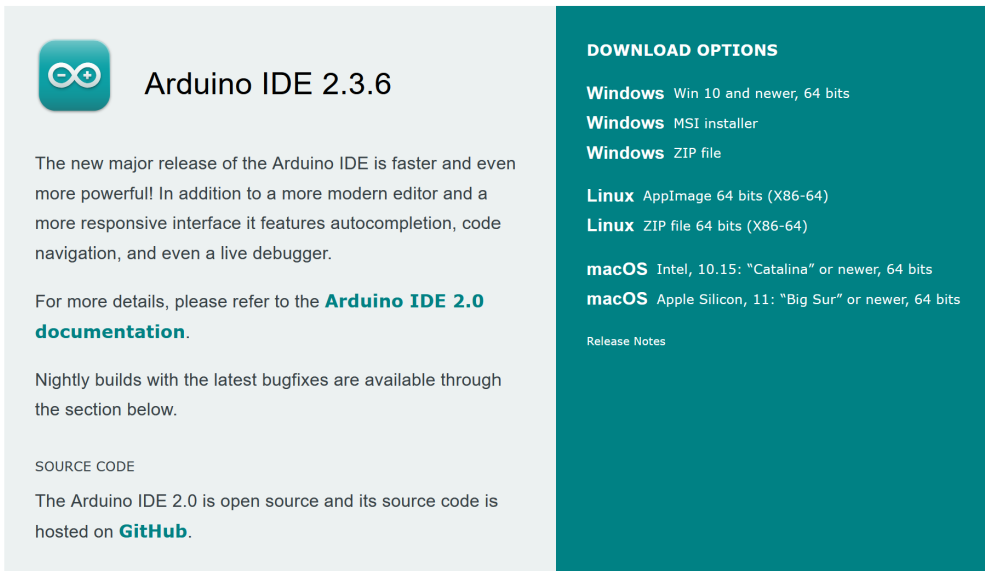
Das Serieller Monitor-Symbol öffnet ein Terminal zur Anzeige und zum Versand von Textdaten über die serielle Verbindung, ermöglicht also die Echtzeit-Kommunikation mit dem Board.

2.2. Installation der Arduino IDE

2.2.1. Download der Arduino IDE

Der Arduino Nano 33 BLE Sense nutzt die Arduino Software Integrated Development Environment (IDE) für die Programmierung, die am weitesten verbreitete IDE für alle Arduino-Boards und sowohl online als auch offline verwendbar ist.
Diese Open-Source-Arduino-IDE vereinfacht das Schreiben von Code und das Hochladen auf das Board. Für jedes Betriebssystem (OS), einschließlich macOS, Linux und Windows, sind verschiedene Versionen der Software verfügbar. Die Arduino-Community bietet zudem eine Online-Plattform zum Programmieren und Speichern von Sketches in der Cloud. Dieser Online-Arduino-Editor ist die neueste Version der IDE und beinhaltet alle Bibliotheken sowie Unterstützung für neue Arduino-Boards.
Um auf diese Softwarepakete zuzugreifen, besuchen Sie die folgende Website: Arduino-Software, um auf dem neuesten Stand zu bleiben, da tägliche Updates über den genannten Link verfügbar sind.
Der Editor kann direkt von der Arduino-Software-Seite heruntergeladen werden. Die Download-Optionen sind in Abbildung 2.2 dargestellt.

Figure 2.2.: Arduino IDE 2.3.6



2.2.2. Installation auf Windows

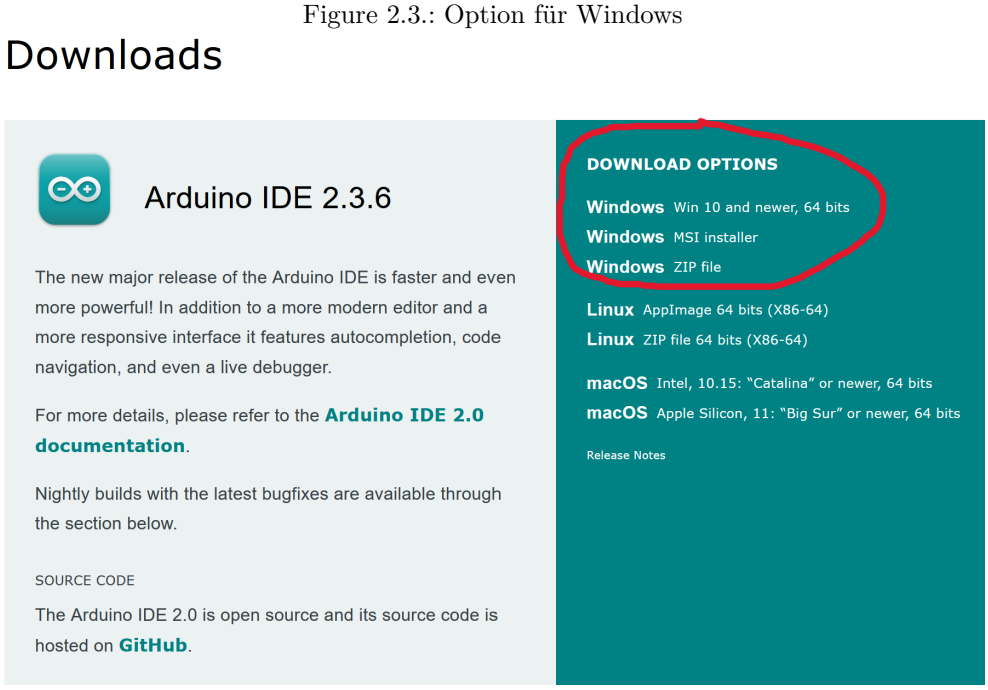
Der Installationsprozess der Arduino IDE 2 auf einem Windows-Computer wird im folgenden Abschnitt Schritt für Schritt beschrieben. So installieren Sie die Arduino

IDE 2 auf einem Windows-Computer:

Führen Sie die von der Software-Seite Arduino-Software heruntergeladene Datei aus.

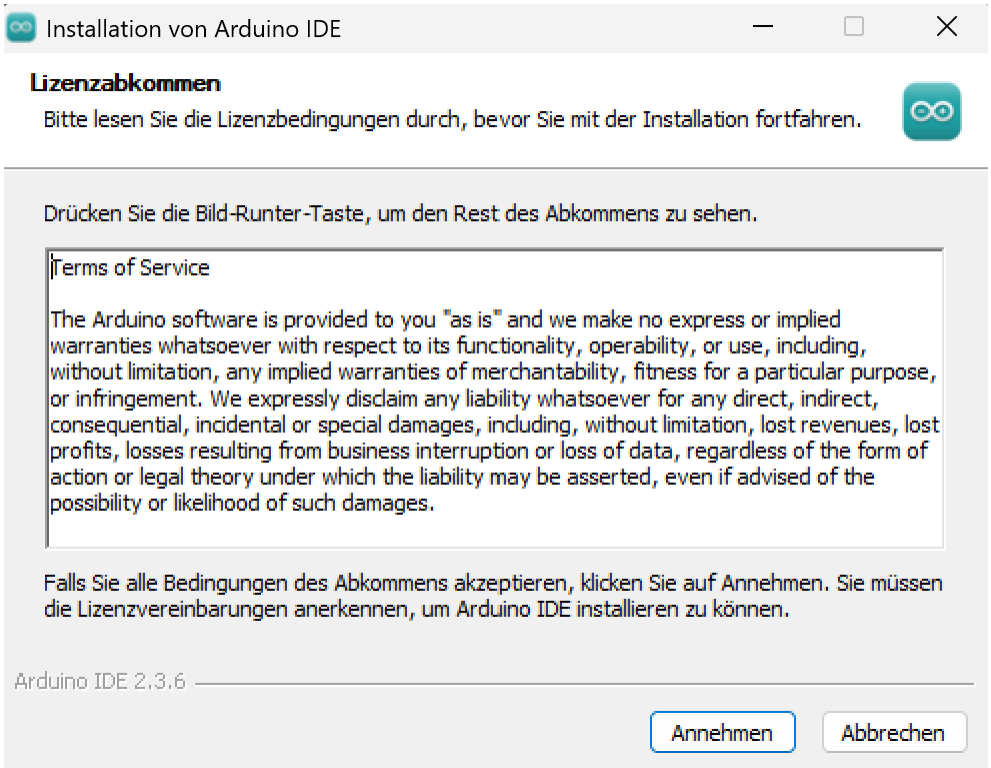
Nutzen Sie die Installationsmöglichkeiten unter Windows, wie in Abbildung 2.3 dargestellt, um eine korrekte Installation sicherzustellen.

Figure 2.3.: Option für Windows



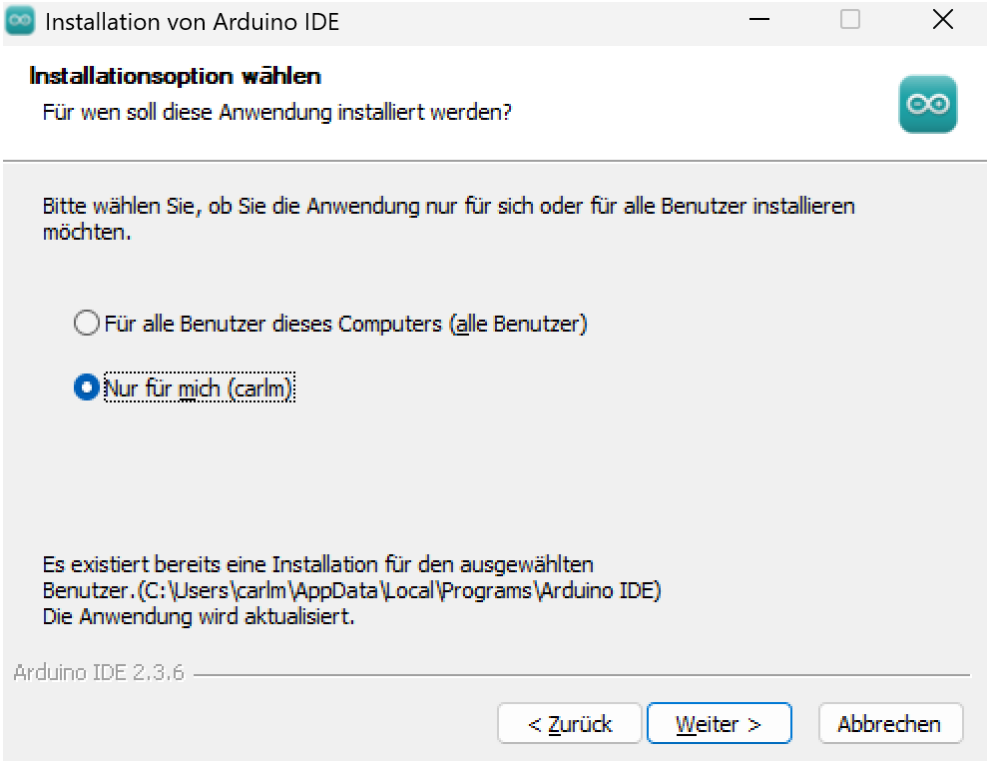
Nachdem der Download abgeschlossen ist, führen Sie die Setup-Datei aus. Akzeptieren Sie das Lizenzabkommen 2.4 und klicken sie auf Annehmen:

Figure 2.4.: Lizenzabkommen



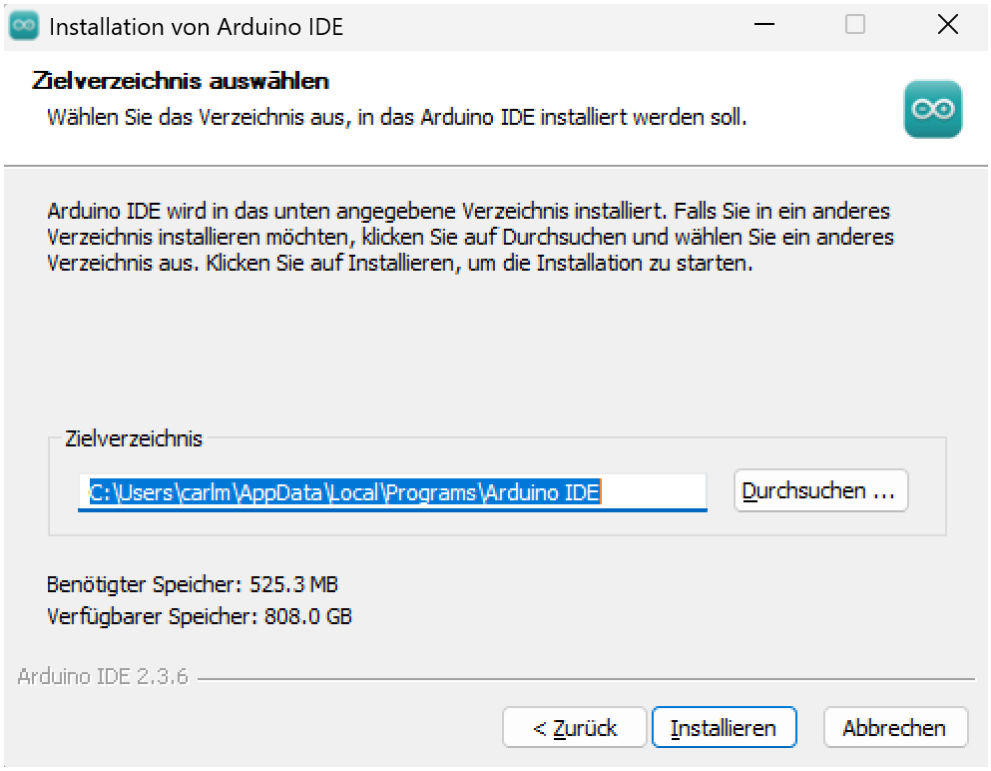
Danach wählen Sie aus, für wen die Installation auf Ihrem PC angelegt werden soll, Sie können auswählen, ob für Sie selbst oder alle Benutzer des Computers: 2.5

Figure 2.5.: Auswahl der Benutzer



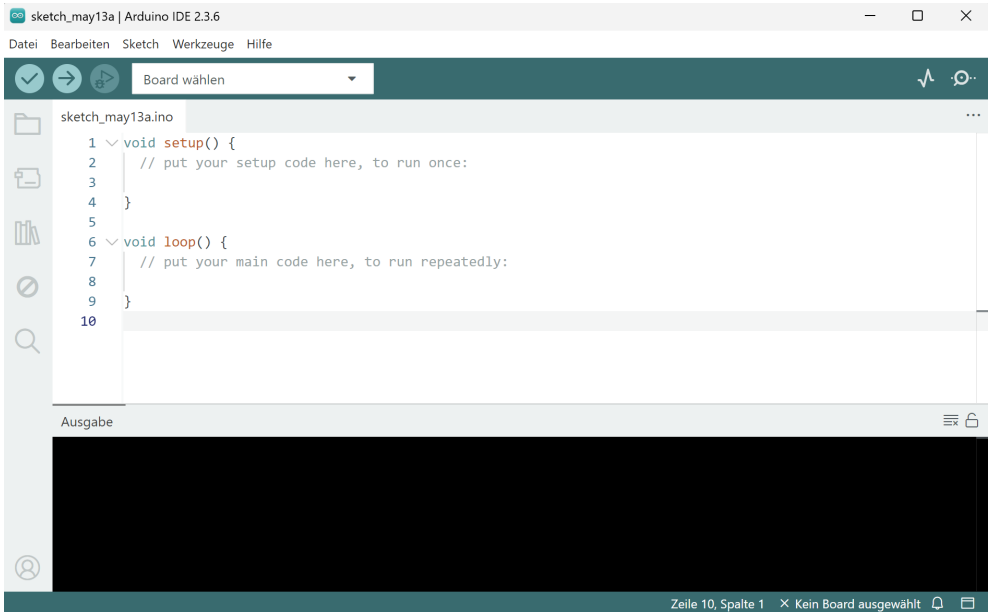
Nachdem Sie Ihre Wahl getroffen haben, klicken Sie auf Weiter. Nun können Sie das Verzeichnis auswählen, in das die Arduino IDE installiert werden soll. 2.6

Figure 2.6.: Auswahl des Installationsverzeichnis



Klicken Sie dann auf Installieren und warten Sie ab, bis der Download abgeschlossen ist. Nach Abschluss des Downloads öffnet sich die Arduino IDE automatisch. 2.7

Figure 2.7.: Die Arduino IDE



2.2.3. Installation (MacOS)

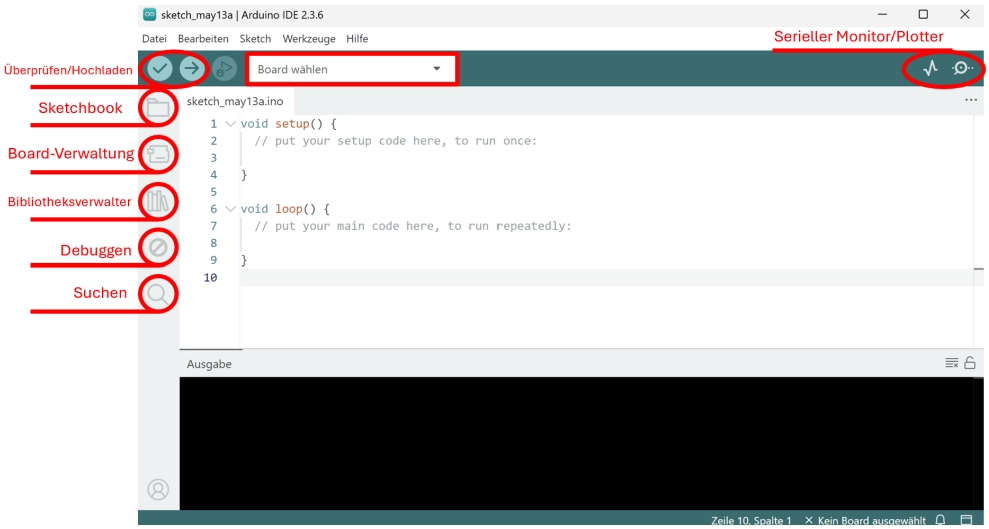
Aufgrund fehlender Hardware können wir leider keine Installation auf MacOS bereitstellen.

2.3. Übersicht

Die Arduino IDE 2 beeinhaltet eine neue Sidebar 2.8, auf der die am häufigsten genutzten Werkzeuge leichter zu erreichen sind. Folgende Werkzeuge sind besonders hilfreich: [Ard25]

- **Überprüfen / Hochladen** Diese Funktionen dienen zum Kompilieren und Hochladen von Code auf ein Arduino-Board.
- **Board wählen** Erkannte Arduino-Boards werden hier automatisch angezeigt, inklusive der Port-Nummer.
- **Sketchbook** Dieser Bereich dient als zentrale Ablage für alle Sketches, die lokal auf dem Computer gespeichert sind. Das Sketchbook bietet eine strukturierte, leicht zugängliche Umgebung zur Verwaltung, Organisation und Bearbeitung von Arduino-Code. Zusätzlich ermöglicht es die Synchronisation mit der Arduino Cloud, wodurch Sketches geräteübergreifend verfügbar sind. Diese Cloud-Integration erlaubt den Zugriff und die Bearbeitung von Sketches aus der Online-Arduino-Umgebung.
- **Board-Verwaltung** Durchsuchen von Arduino- und Drittanbieter-Paketen zur Installation. Beispielsweise erfordert ein MKR WiFi 1010 Board das Arduino SAMD Boards-Paket.
- **Bibliotheksverwalter** Zugriff auf tausende Arduino-Bibliotheken, bereitgestellt von Arduino und der Community.
- **Debuggen** Echtzeit-Test und Fehlersuche in Programmen.
- **Suchen** Durchsucht den geschriebenen Code nach Schlüsselwörtern.
- **Serieller Monitor** Öffnet den seriellen Monitor als neuen Tab in der Konsole.
- **Serieller Plotter** Visualisiert serielle Daten als Echtzeit-Diagramme.

Figure 2.8.: Die neue Arduino IDE mit der nützlichen Sidebar



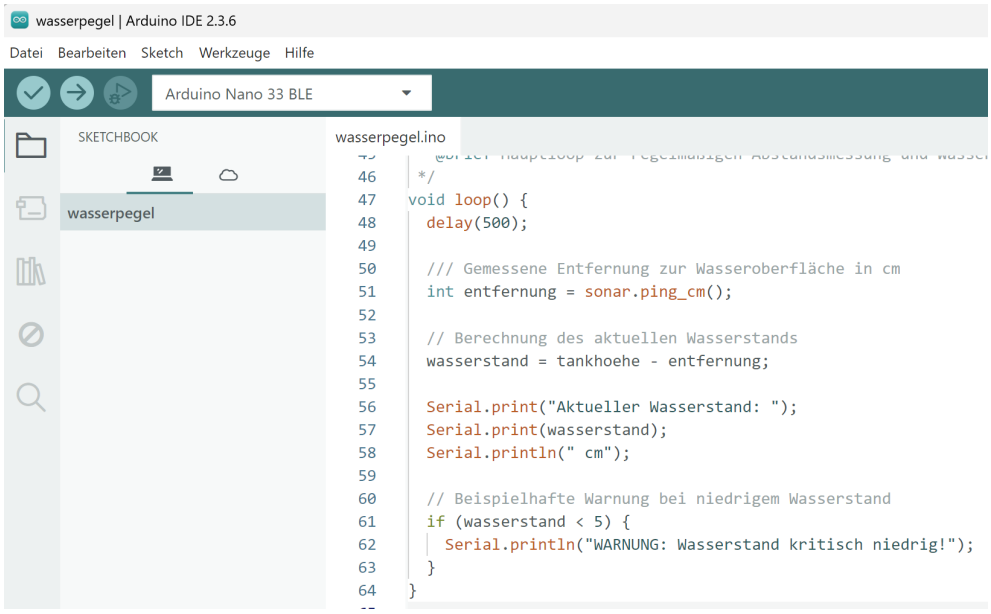
2.4. Funktionen

Die Arduino IDE 2 ist ein vielseitiges Entwicklungswerkzeug, das Funktionen wie die direkte Installation von Bibliotheken, Cloud-Synchronisation für Sketches und integrierte Debugging-Tools bietet. Dieser Abschnitt beleuchtet einige der Kernfunktionen, mit Verweisen auf detailliertere Ressourcen.

2.4.1. Sketchbook

Das **Sketchbook** ist der Speicherort für Arduino-Codedateien, die als Sketches bezeichnet werden. Diese Dateien werden mit der Endung `.ino` gespeichert. Zur ordnungsgemäßen Organisation muss jeder Sketch in einem Ordner abgelegt werden, der genau denselben Namen trägt wie die Sketch-Datei. Beispielsweise muss ein Sketch mit dem Namen `MySketch.ino` in einem Ordner namens `MySketch` gespeichert werden. Diese Namenskonvention ist entscheidend, damit die Arduino IDE die Sketches korrekt identifizieren und verwalten kann.
Standardmäßig werden Sketches in einem Ordner namens `Arduino` gespeichert, der sich im Dokumentenverzeichnis des Systems befindet.
Um auf das Sketchbook zuzugreifen, können Sie das Ordnersymbol in der Seitenleiste der Arduino IDE auswählen. Diese Aktion öffnet das Verzeichnis, in dem die Sketches gespeichert sind, und ermöglicht so eine effiziente Verwaltung und den Zugriff auf die Sketches. 2.9

Figure 2.9.: Das Sketchbook auf der Arduino IDE



2.4.2. Board-Verwaltung

Die Board-Verwaltung kann unter folgendem Pfad gefunden werden: Werkzeuge > Board > Board-Verwaltung...
Der **Boards Manager** in der Arduino IDE ermöglicht die Installation verschiedener Board-Pakete, wie in Abbildung 2.10 dargestellt. Ein Board-Paket enthält die erforderlichen Dateien und Anleitungen, um Code für bestimmte Arduino-Board-Typen zu kompilieren und hochzuladen.
Es stehen verschiedene Board-Pakete zur Verfügung, darunter:

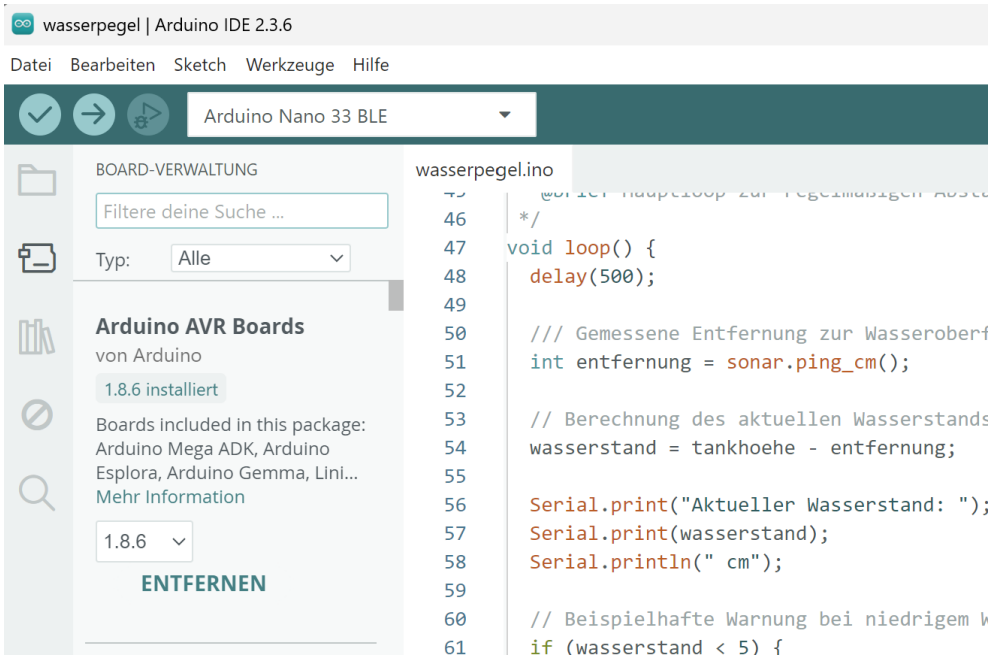
- **avr** - für ältere Boards wie das Arduino Uno

- **samd** - für neuere Boards wie das Arduino Zero

- **megaavr** - für die Mega-Serie

Jedes Paket unterstützt unterschiedliche Arduino-Board-Familien. Der Boards Manager stellt sicher, dass die richtigen Tools und Dateien für die Programmierung des ausgewählten Boards installiert sind.

Figure 2.10.: Die Board-Verwaltung auf der Arduino IDE



2.4.3. Bibliotheksverwalter

Den **Bibliotheksverwalter** erreichen Sie über folgenden Pfad:
Werkzeuge > Bibliotheken verwalten...

Der Bibliotheksverwalter in der Arduino IDE ermöglicht das Durchsuchen und Installieren einer Vielzahl von Bibliotheken. Diese Bibliotheken erweitern die Grundfunktionen von Arduino und vereinfachen die Arbeit mit speziellen Hardwarekomponenten, wie zum Beispiel:

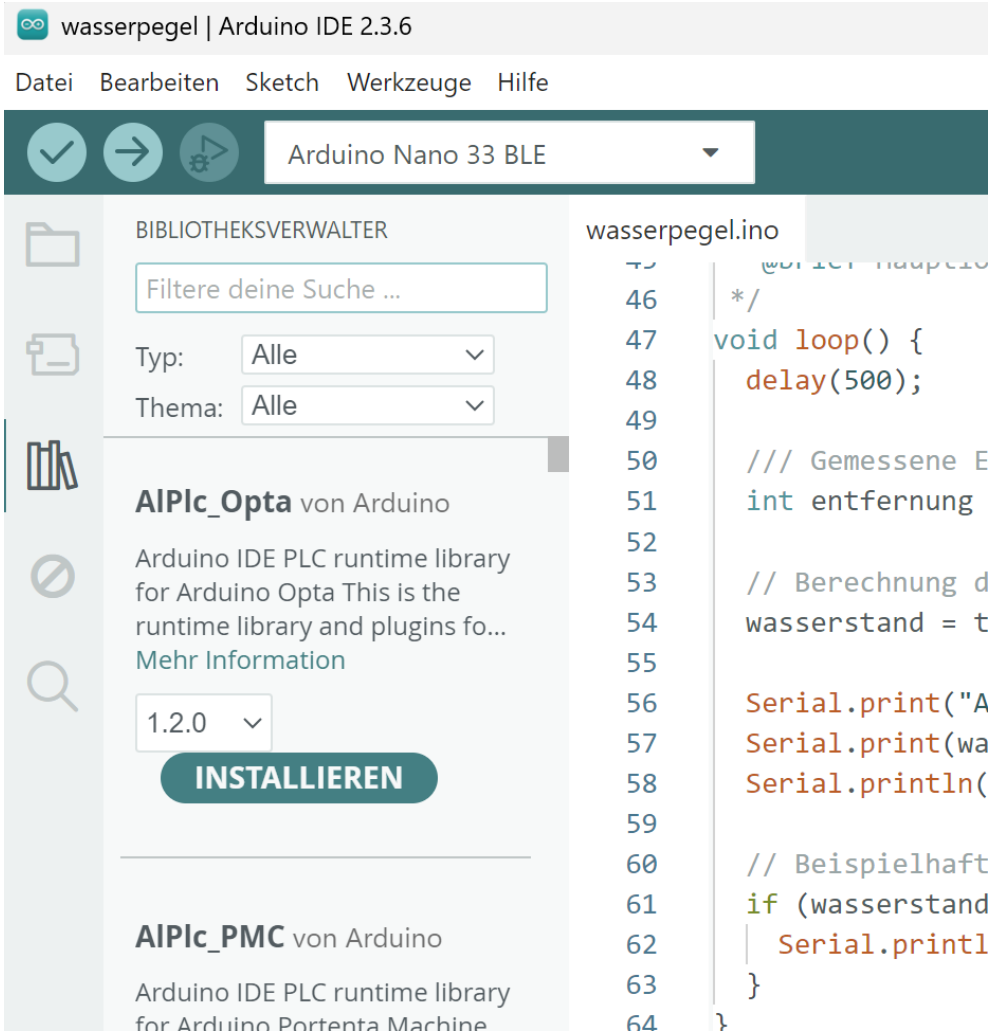
- Steuerung von Servomotoren

- Auslesen verschiedener Sensoren

- Kommunikation mit Modulen (z.B. Wi-Fi)

Die Bibliotheken enthalten vorgefertigten Code zur Ansteuerung spezifischer Hardware und vereinfachen komplexe Aufgabenstellungen. Dies ermöglicht eine schnellere und effizientere Entwicklung von Arduino-Projekten, wie in Abbildung 2.11 dargestellt.

Figure 2.11.: Der Bibliotheksverwalter auf der Arduino IDE



2.5. Integration und Verwendung einer benutzerdefinierten Bibliothek in der Arduino IDE

Bei der Entwicklung eingebetteter Systeme mit der Arduino-Plattform spielen Bibliotheken eine zentrale Rolle, um komplexe Funktionalitäten zu abstrahieren und zu vereinfachen. Eine benutzerdefinierte Bibliothek ist besonders vorteilhaft, wenn bestimmte Funktionen oder Hardware-Schnittstellen in verschiedenen Projekten wiederholt verwendet werden. Dieser Abschnitt beschreibt den Prozess der Erstellung, Installation und Verwendung einer benutzerdefinierten Bibliothek innerhalb eines Arduino-Projekts in der Arduino Integrated Development Environment (IDE).

2.5.1. Erstellung der Bibliotheksdateien

Eine gut strukturierte benutzerdefinierte Arduino-Bibliothek besteht typischerweise aus mindestens zwei Schlüsselkomponenten:
Header-Datei (.h): Diese Datei enthält die Deklarationen von Klassen, Funktionen und Variablen, welche die Schnittstelle der Bibliothek definieren. Sie stellt die notwendigen Definitionen bereit, damit die Funktionen und Klassen von externen Programmen

genutzt werden können.
Quellcode-Datei (.cpp): Diese Datei enthält die eigentliche Implementierung der in der Header-Datei deklarierten Funktionen und Methoden. Sie definiert das Verhalten der Funktionen und Klassen.
Die Bibliotheksstruktur ist so organisiert, dass sie eine klare Trennung zwischen Deklaration und Implementierung der Funktionalitäten ermöglicht.

2.5.2. Installation und Integration der Bibliothek

Nach der Erstellung der benutzerdefinierten Bibliothek muss diese korrekt in der Arduino IDE installiert und integriert werden, um in Projekten verwendet werden zu können.
Gehen Sie folgendermaßen vor, um eine benutzerdefinierte Bibliothek in Arduino zu installieren:

1. **Bibliotheksordner lokalisieren**: Die Arduino IDE sucht Bibliotheken im `Bibliotheken`-Ordner, der sich im Arduino-Sketchbook-Verzeichnis befindet. Der typische Pfad lautet:

   ```
   C:/Benutzer/<Benutzername>/Dokumente/Arduino/Bibliotheken
   ```

2. **Bibliothek kopieren**: Kopieren Sie den Ordner der benutzerdefinierten Bibliothek (z.B. `Nano33BLESenseLED`) in das `Bibliotheken`-Verzeichnis. Der vollständige Pfad sollte wie folgt aussehen:

   ```
   C:/Benutzer/<Benutzername>/Dokumente/Arduino/Bibliotheken/Nano33BLESenseLED
   ```

3. **Arduino IDE neu starten**: Nachdem die Bibliothek im richtigen Verzeichnis platziert wurde, starten Sie die Arduino IDE neu, damit die neue Bibliothek erkannt wird.

2.5.3. Verwendung symbolischer Links zur Bibliotheksintegration

In bestimmten Fällen kann es praktischer oder notwendig sein, die Bibliotheksdateien außerhalb des Standard-Bibliotheksverzeichnisses zu speichern. Dies ist beispielsweise sinnvoll, wenn:

- Die Bibliotheken in einem GitHub-Repository verwaltet werden
- Eine bessere Versionskontrolle benötigt wird
- Die Bibliotheken mit mehreren Projekten geteilt werden sollen

In solchen Fällen kann der Befehl `mklink` verwendet werden, um einen symbolischen Link zu erstellen. Dieser Link ermöglicht der Arduino IDE den Zugriff auf die Bibliotheksdateien, als wären sie im Standardverzeichnis gespeichert, ohne dass die Dateien tatsächlich dupliziert werden müssen.
Vorgehensweise für Windows:

1. Öffnen Sie die Eingabeaufforderung als Administrator
2. Führen Sie folgenden Befehl aus:

   ```
   mklink /D
   "Pfad im Arduino Verzeichnis"
   "Original Bibliothekspfad"
   ```

Vorteile dieser Methode:

- Zentrale Verwaltung der Bibliotheksversionen
- Keine redundante Speicherung der Dateien
- Automatische Synchronisation zwischen Projekten
- Einfache Aktualisierung der Bibliotheken

2.5.4. Windows-Tool `mklink` zur Erstellung symbolischer Links

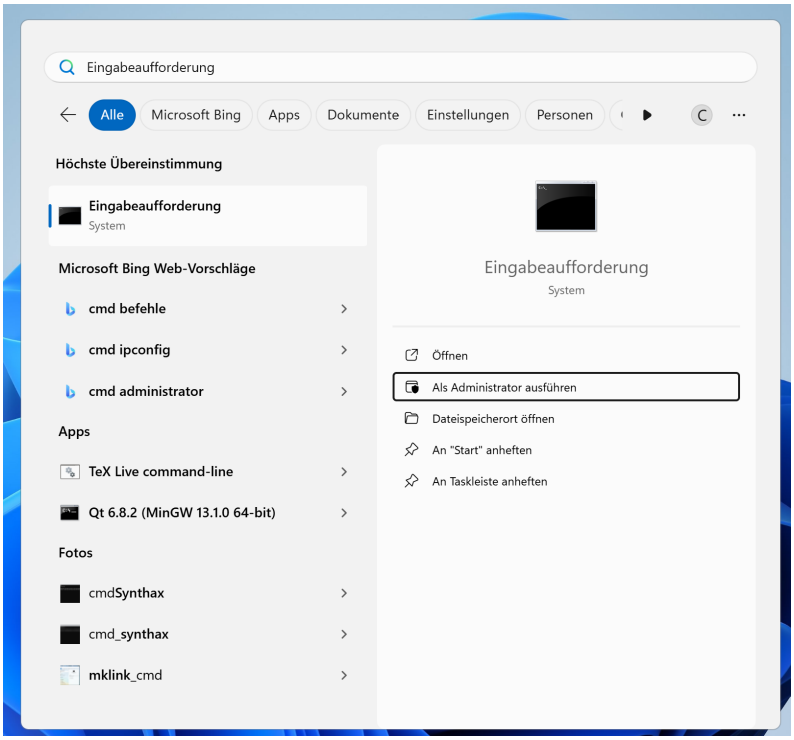
Symbolische Links ermöglichen die Speicherung von Bibliotheken an einem anderen Ort, während die Arduino IDE sie weiterhin im Standardverzeichnis `libraries` behandelt. Dies ist besonders nützlich für:

- Versionskontrollsysteme wie Git
- Organisation von Bibliotheken ohne Dateiduplizierung
- Geteilte Bibliotheken zwischen mehreren Projekten

Schritte zur Erstellung eines symbolischen Links:

1. **Eingabeaufforderung mit Administratorrechten öffnen:**
   - Drücken Sie Win + S
   - Suchen Sie nach cmd
   - Klicken Sie auf Eingabeaufforderung
   - Wählen Sie Als Administrator ausführen (siehe Abbildung 2.12)

Figure 2.12.: Administrator-Eingabeaufforderung in Windows



Führe `mklink` aus: Die Eingabeaufforderung für die Erstellung eines symbolischen Links ist: `mklink /D "TargetPath" "SourcePath"`
Zielverzeichnis: Das Verzeichnis, bei dem die Arduino IDE die Bibliothek erwartet: `Bibliotheken`.
Für dieses Beispiel ist die Bibliothek hier:
`C:/Benutzer/<Benutzername>/Dokumente/Desktop/Nano33BLESenseLED`

Das Zielverzeichnis für die den Bibliotheksordner ist:
`C:/Benutzer/<Benutzername>/Dokumente/Arduino/Bibliotheken/Nano33BLESenseLED`

2.5.5. Überprüfung der Bibliotheksverfügbarkeit in der Arduino IDE

Um sicherzustellen, dass die benutzerdefinierte Bibliothek erfolgreich integriert wurde und von der Arduino IDE erkannt wird, gehen Sie wie folgt vor:

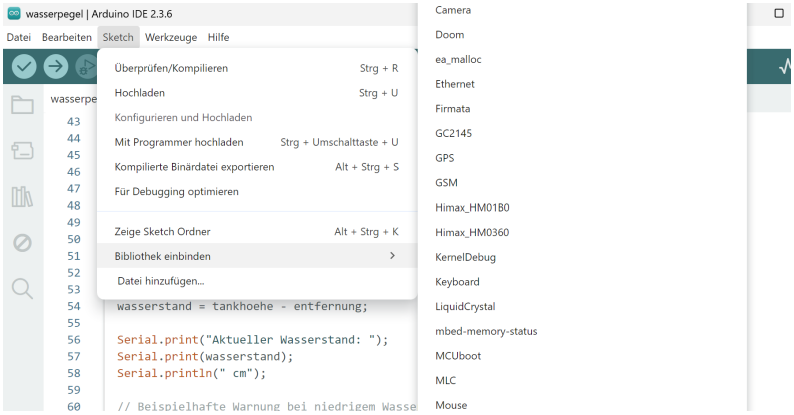
1. **Arduino IDE neu starten**: Falls die IDE während der Installation geöffnet war, schließen und öffnen Sie sie erneut. Dadurch wird die Liste der verfügbaren Bibliotheken aktualisiert.

2. **Bibliotheksmenü prüfen**: Navigieren Sie zu Sketch > Bibliothek einbinden und wählen Sie die Bibliothek `Nano33BLESenseLED` aus (siehe Abbildung 2.13).

3. **Listenpräsenz bestätigen**: Erscheint die Bibliothek in der Liste, wurde sie erfolgreich erkannt. Andernfalls überprüfen Sie die Verzeichnisstruktur und den symbolischen Link.

4. **Testprogramm kompilieren**: Erstellen Sie ein einfaches Testprogramm mit Funktionen der Bibliothek. Ein erfolgreicher Kompiliervorgang bestätigt die korrekte Integration.

Figure 2.13.: Einbindung der Nano33BLESenseLED-Bibliothek



Nach erfolgreicher Kompilierung ist die Bibliothek einsatzbereit und kann in Arduino-Projekten verwendet werden.

2.5.6. Serieller Monitor

Der serielle Monitor kann über folgenden Pfad aufgerufen werden:
Werkzeuge > Serieller Monitor
Dieses Tool ermöglicht die Anzeige von Daten, die vom Board gesendet werden, beispielsweise über den Befehl `Serial.print()`.

2.5.7. Serieller Plotter

Der serielle Plotter ist ein nützliches Werkzeug zur Visualisierung von Daten in Form von Graphen und eignet sich besonders gut für:

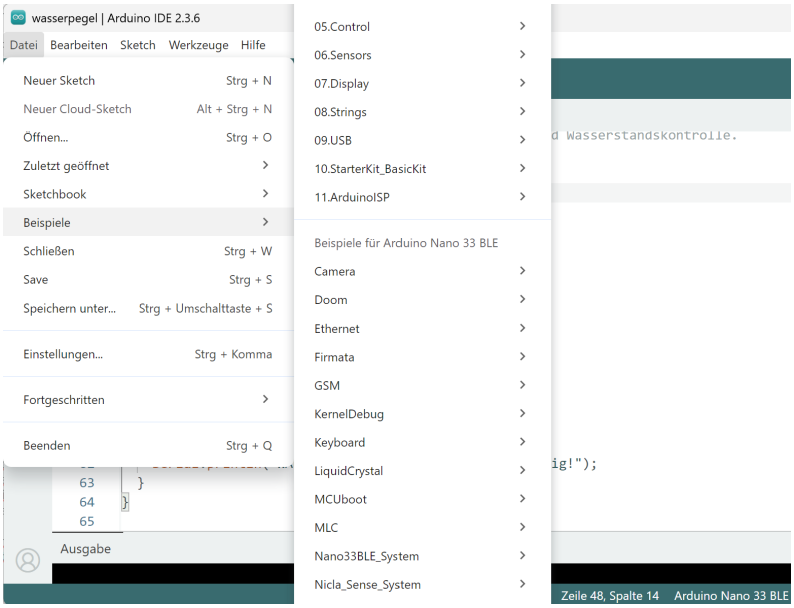
- Echtzeit-Monitoring von Sensorwerten
- Analyse von Spannungsspitzen und Signalverläufen
- Gleichzeitige Darstellung mehrerer Variablen
- Identifikation von Trends und Mustern in Datenströmen

Er kann über den folgenden Pfad aufgerufen werden: Werkzeuge > Serieller Plotter

2.6. Beispiel-Sketches

Ein wesentlicher Bestandteil der Arduino-Dokumentation sind die mit verschiedenen Bibliotheken mitgelieferten Beispiel-Sketches. Diese Beispiele veranschaulichen die praktische Anwendung von Bibliotheksfunktionen und zeigen deren vorgesehene Verwendungszwecke sowie Hauptmerkmale auf. Bibliotheken, die mit bestimmten Board-Paketen geliefert werden, können ebenfalls eigene Beispielsketche enthalten.
Um auf diese Beispielsketche zuzugreifen - sowohl für manuell installierte Bibliotheken als auch für in Board-Paketen enthaltene - navigieren Sie zu: 2.14 Datei > Beispiele und wählen Sie die gewünschte Bibliothek aus der Liste aus.

Figure 2.14.: Beispielsketche in der Arduino IDE



2.7. Debugging

Der Debugger in der Arduino IDE 2.3.6 ist ein leistungsfähiges Werkzeug zur Fehlersuche und Programmoptimierung. Er ermöglicht eine kontrollierte Ausführung des Programmcodes und bietet umfangreiche Analysefunktionen.

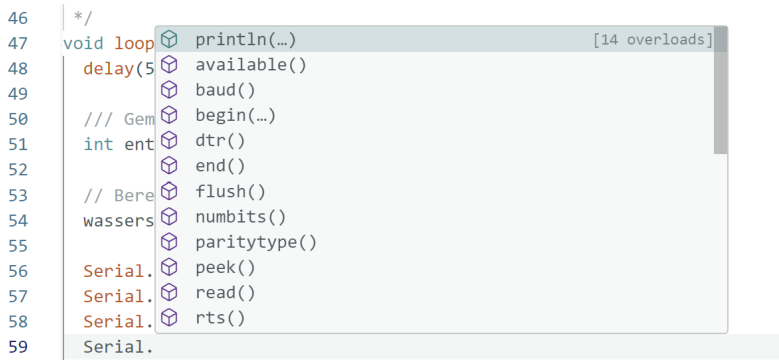
Funktionsumfang

- **Breakpoints**: Markieren Sie bestimmte Codezeilen, an denen die Ausführung automatisch pausieren soll
- **Einzelschrittmodus**: Führen Sie das Programm zeilenweise aus (Step Over/Step Into)
- **Variableninspektion**: Überwachen Sie Werte und Zustände von Variablen in Echtzeit
- **Aufrufstack-Analyse**: Verfolgen Sie den Programmfluss durch verschiedene Funktionsebenen

2.8. Autovervollständigung

Die Autovervollständigung ist eine Schlüsselfunktion der Arduino IDE Version 2, die das Programmieren durch intelligente Vorschläge für Funktionen und Elemente der Arduino-API wesentlich vereinfacht. Diese Funktion unterstützt Entwickler bei der schnelleren und präziseren Code-Erstellung. 2.15

Figure 2.15.: Autovervollständigen in der Arduino-IDE



2.9. Zusammenfassung

Diese Anleitung bietet einen umfassenden Überblick über die wesentlichen Funktionen der Arduino IDE 2 und verweist auf detaillierte Artikel für vertiefende Informationen. Die einzelnen Abschnitte sind darauf ausgerichtet, das Verständnis und die effektive Nutzung der vielfältigen Möglichkeiten der Entwicklungsumgebung zu fördern.

Wesentliche Erkenntnisse

- Die Arduino IDE 2 vereint alle wichtigen Werkzeuge für die Arduino-Entwicklung in einer modernen Oberfläche
- Neue Funktionen wie der integrierte Debugger und verbesserte Code-Vervollständigung beschleunigen den Entwicklungsprozess
- Die Cloud-Integration ermöglicht flexible Arbeitsabläufe über mehrere Geräte hinweg
- Umfangreiche Dokumentation und Beispielsketche erleichtern den Einstieg

Empfehlungen für Nutzer

- Regelmäßige Updates der IDE durchführen
- Die offizielle Dokumentation für fortgeschrittene Themen nutzen
- Community-Beispiele als Lerngrundlage verwenden
- Debugging-Funktionen für komplexe Projekte einsetzen

Die Arduino IDE 2 stellt eine leistungsfähige Plattform für Einsteiger und erfahrene Entwickler dar, die kontinuierlich weiterentwickelt wird. Durch die systematische Nutzung der beschriebenen Funktionen können Projekte effizienter realisiert und Probleme schneller gelöst werden.

3. Einrichtung des Arduino Nano 33 BLE Sense

3.1. Einführung

In diesem Kapitel wird der Prozess der Verbindung des Arduino Nano 33 BLE Sense mit einem Computer oder Laptop sowie die Konfiguration der erforderlichen Grundeinstellungen beschrieben. Die folgenden Schritte werden detailliert erläutert:

- Installation der notwendigen Software-Tools
- Korrekte Einrichtung des Boards in der Arduino IDE
- Verifizierung der Funktionsfähigkeit durch ein Beispielsketch

3.2. Konfiguration des Arduino Nano 33 BLE Sense

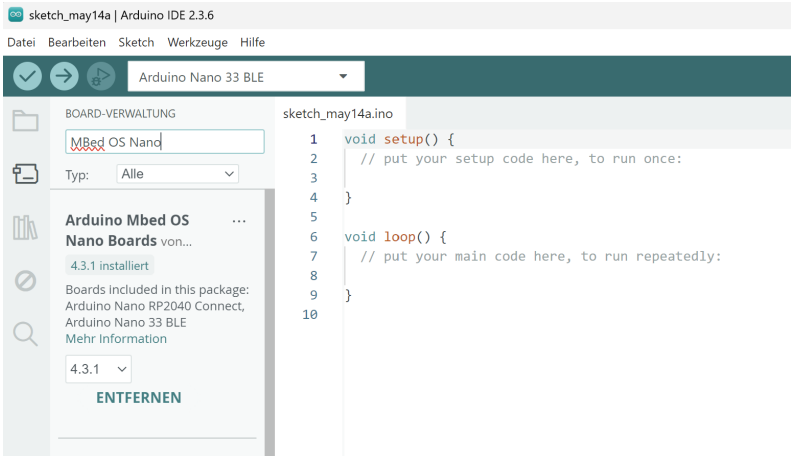
Um den **Arduino Nano 33 BLE Sense** in einer Offline-Umgebung zu programmieren, gehen Sie wie folgt vor:

Installation des Treibers

1. **Installation der Arduino IDE:** Installieren Sie zunächst die neueste Version der Arduino IDE auf Ihrem Computer.
2. **Installation der Pakete:** Nach Abschluss der IDE-Installation öffnen Sie die Arduino IDE und navigieren zum Menü Werkzeuge in der oberen linken Ecke.
3. Wählen Sie im Menü Werkzeuge den Board-Verwalter aus.
4. Im Fenster Board-Verwalter verwenden Sie die Suchleiste, um den **Arduino Nano 33 BLE Sense** zu finden, indem Sie den Namen eingeben (siehe Abbildung 3.1).
5. Wählen Sie aus den Suchergebnissen Arduino Mbed OS Nano Boards aus und klicken Sie auf Installieren, um das benötigte Paket zu installieren.

Das Mbed OS Nano Board-Paket beinhaltet die Unterstützung für mehrere Boards der Arduino Nano Familie, einschließlich des Arduino Nano 33 BLE Sense. Nach Abschluss der Installation verbinden Sie den Arduino Nano 33 BLE Sense über ein USB-A zu USB-Micro Kabel mit Ihrem Computer, um mit der Programmierung zu beginnen.

Figure 3.1.: Mbed OS Nano-Paket installation



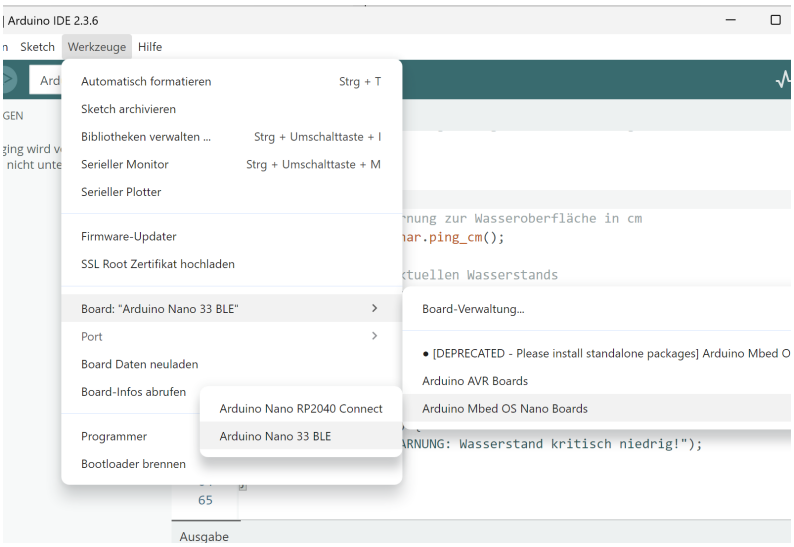
3.2.1. Verbindung und Konfiguration des Arduino-Boards mit dem Computer

Um entweder ein integriertes Beispiel oder ein eigenes Programm auf dem Arduino Nano 33 BLE Sense auszuführen, gehen Sie wie folgt vor:

1. Verbinden Sie das Arduino-Board über ein USB-A-zu-USB-Micro-Kabel mit dem Computer.
2. Öffnen Sie die Arduino IDE auf dem Computer. Es erscheint eine leere Arbeitsumgebung mit den Standardfunktionen `void setup()` und `void loop()`.
3. Navigieren Sie zum Menü Werkzeuge, wählen Sie Board und selektieren Sie das angeschlossene Board **Arduino Nano 33 BLE Sense** (siehe Abbildung 3.2).
4. Wählen Sie anschließend den korrekten Port unter Werkzeuge > Port aus.

Diese Einrichtung ermöglicht der Arduino IDE, das Arduino Nano 33 BLE Sense zu erkennen und mit ihm zu kommunizieren.

Figure 3.2.: Verbinden des Mbed OS Nano-Pakets mit dem Board



3.2.2. Auswahl des richtigen Ports

Nach der Auswahl des Arduino Nano 33 BLE Sense Boards muss der korrekte Kommunikationsport eingestellt werden. Gehen Sie hierfür wie folgt vor: **Bootloader-Modus aktivieren:**

1. Drücken Sie den weißen Reset-Knopf auf dem Board (siehe Abbildung 3.4)
2. Der Knopf befindet sich in der Nähe des USB-Anschlusses
3. Die Onboard-LED blinkt während des Bootloader-Modus

Port auswählen:

1. Navigieren Sie in der Arduino IDE zu Werkzeuge > Port
2. Wählen Sie den Port mit der Bezeichnung Arduino Nano 33 BLE Sense aus
3. Unter Windows erscheint dieser typischerweise als COM-Port (z.B. COM3)

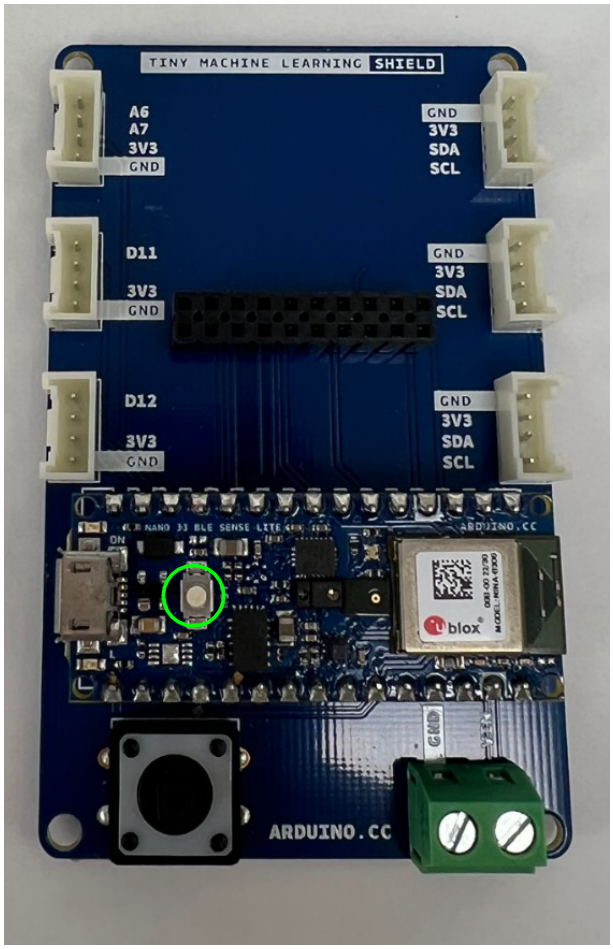


Figure 3.3.: Arduino Nano 33 BLE Sense Reset Button

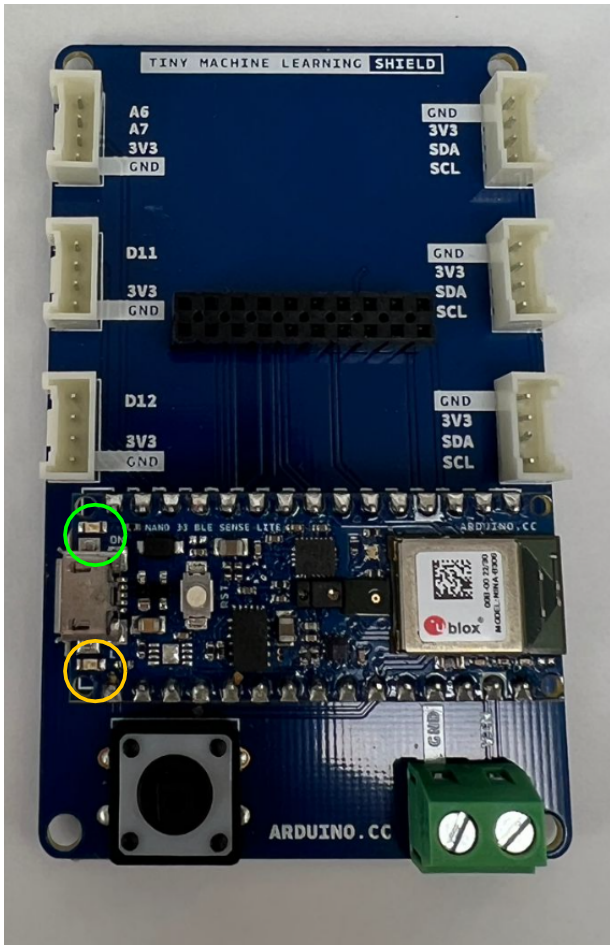


Figure 3.4.: green and orange Builtin-LED

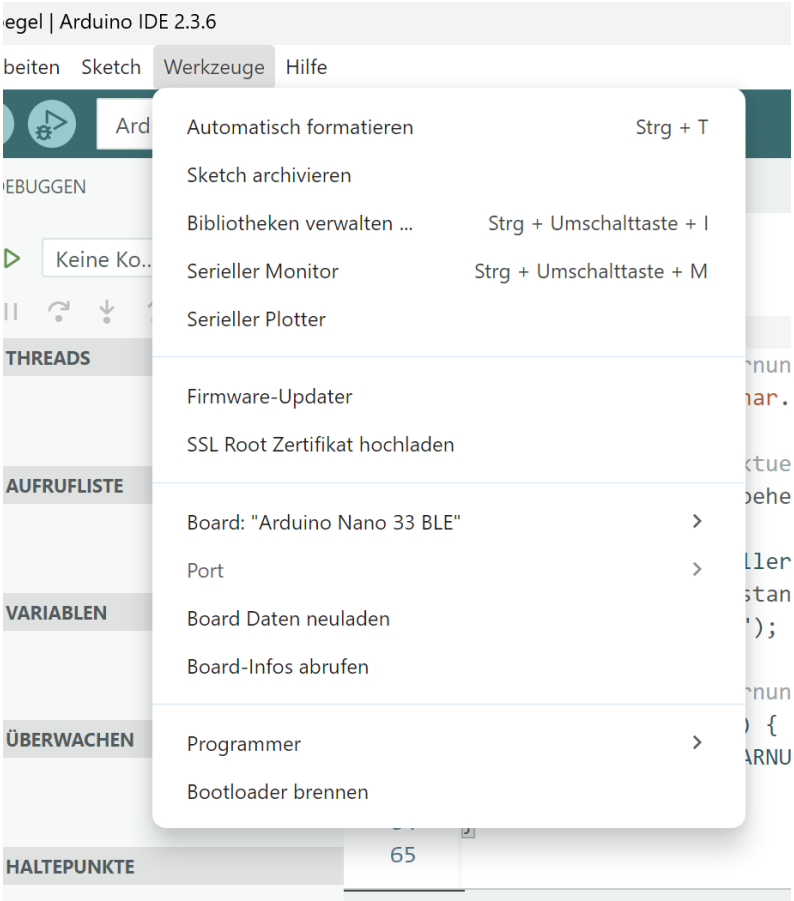
Nach erfolgreicher Durchführung der vorherigen Schritte muss der korrekte Port für den Upload des Programms ausgewählt werden:

1. Navigieren Sie zum Menü Werkzeuge

2. Wählen Sie den Unterpunkt Port aus

3. Selektieren Sie den verfügbaren Port, der mit Ihrem Arduino Nano 33 BLE Sense verbunden ist, wie in 3.5

Figure 3.5.: Selektrieren des verfügbaren Ports



3.2.3. Sketch verifizieren und hochladen

Nach der Auswahl des korrekten Ports folgt der Upload des Arduino-Programms. Gemäß Best Practices sollte das Programm **zuerst verifiziert** werden:

1. Klicken Sie auf Überprüfen
2. Die IDE prüft nun auf Syntaxfehler und gibt Warnungen aus
3. Bei erfolgreicher Verifizierung klicken Sie auf Hochladen

Verbindungsstatus:

- Der korrekte Verbindungsstatus wird in der unteren rechten Ecke der IDE angezeigt
- Dort sollten Board-Typ (Arduino Nano 33 BLE Sense) und Port aufgeführt sein

Upload-Prozess:

- Der Compilierstatus wird im schwarzen Ausgabefenster angezeigt
- Nach erfolgreichem Upload blinkt die Onboard-LED (Pin 13) kurz auf
- Bei Port-Problemen: Port erneut unter Werkzeuge > Port auswählen

3.3. Konfiguration testen

Zur Überprüfung der korrekten Einrichtung führen wir den Beispiel-Sketch Blink aus:

1. Öffnen Sie Datei > Beispiele > 01.Basics > Blink
2. Klicken Sie auf Hochladen
3. Die orangefarbene Onboard-LED (Pin 13) sollte nun im 1-Sekunden-Takt blinken

3.3.1. Test der Einrichtung mit dem Beispiel-Sketch `blink.ino`

Die Arduino IDE bietet zahlreiche Beispiele zum Testen der Hardware. Zur Überprüfung der korrekten Konfiguration eignet sich besonders das grundlegende Beispiel `blink.ino`, das nach dem Upload die Onboard-LED mit 1 Hz blinken lässt.
Vorgehensweise:

1. **Beispiele öffnen:**
   - Arduino IDE starten
   - Im Menü Datei auf Beispiele zeigen
2. **Beispielsketch auswählen:**
   - Kategorie 01.Basics auswählen
   - Sketch Blink öffnen (siehe Abbildung 3.6)
3. **Blink-Sketch analysieren:**
   - Der geöffnete Sketch erscheint in neuem Fenster (Abbildung 3.7)
   - Der Code blinkt die LED an Pin 13 im Sekundentakt
4. **Sketch uploaden:**
   - Auf Hochladen klicken (Pfeil-Symbol)
   - Nach erfolgreichem Upload blinkt die orange Onboard-LED

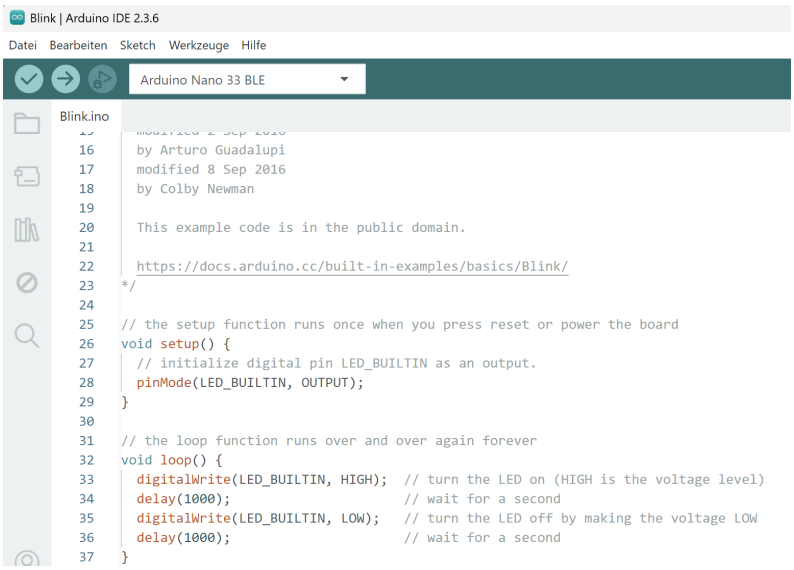


Figure 3.6.: Navigation zum Blink-Beispiel in der Arduino IDE

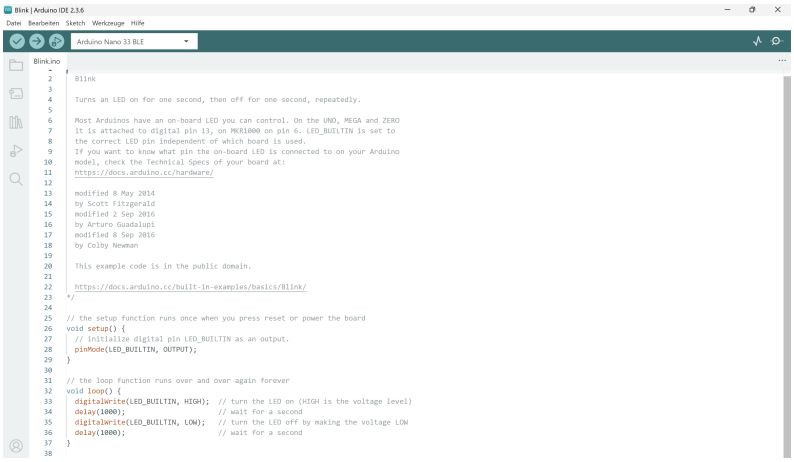


Figure 3.7.: Code des Blink-Beispiels in der Arduino IDE

Erfolgskontrolle:

- Die orange LED (Pin 13) blinkt im 1-Sekunden-Takt
- Im Ausgabefenster erscheint "Upload abgeschlossen"
- Keine Fehlermeldungen im schwarzen Konsolenfenster

Damit ist die Funktionsfähigkeit des Arduino Nano 33 BLE Sense bestätigt und das Board kann für anspruchsvollere Projekte verwendet werden

4. Arduino Nano 33 BLE Sense

Arduino ist eine Open-Source-Elektronikplattform, die auf flexibler und benutzerfreundlicher Hardware sowie Software basiert. Sie bietet Mikrocontroller- und Mikroprozessor-Boards zur Entwicklung digitaler und analoger Geräte.
Der Mikrocontroller - oft als "Mini-Computer" des Arduino-Boards bezeichnet - benötigt normalerweise verschiedene elektronische Bauteile wie Dioden, Widerstände, Kondensatoren und Transistoren zur Spannungs- und Stromregulierung. Das Arduino-Team hat jedoch eine benutzerfreundliche Umgebung geschaffen, die diese elektronischen Komplexitäten überflüssig macht:

- Einfache Stromversorgung nach Spannungsvorgabe
- Programmierung in wenigen Sekunden
- Automatische Hardware-Steuerung
- Keine zusätzlichen elektronischen Komponenten erforderlich

Durch Fortschritte in der Halbleiter- und Elektronikindustrie werden Steuerungsprobleme heute zunehmend mit diesen kompakten Mikrocontrollern gelöst, anstatt mit mechanischen oder elektrischen Schaltern.[Ard21]

4.1. Arduino Nano 33 BLE Sense

Die Arduino Nano-Familie umfasst kompakte Boards mit umfangreichen Funktionen – vom preisgünstigen Einsteigermodell Nano bis hin zum funktionsreichen Nano BLE Sense/Nano RP2040 Connect mit Bluetooth-/Wi-Fi-Modulen. Diese Boards verfügen über integrierte Sensoren für Temperatur, Luftfeuchtigkeit, Druck, Gestenerkennung, Mikrofon und mehr. Sie unterstützen zudem MicroPython und Machine Learning [Raj19].
Das Arduino Nano 33 BLE Sense ist ein kompaktes Board der Nano-Familie mit:

- Bluetooth Low Energy (BLE 5.0) über das NINA-B306-Modul
- Leistungsstarkem Nordic nRF52480-Prozessor (Cortex M4F)
- Umfangreicher Sensorik:
  - 9-Achsen-IMU (Beschleunigungssensor, Gyroskop, Magnetometer)
  - Umwelt-Sensoren (Temperatur, Luftfeuchtigkeit, Druck)
  - Optische Sensoren (RGB-Licht, Gesten, Näherung)
  - Digitales Mikrofon
- Energieeffizientem Design im Nano-Format (45x18mm)

Namenskonvention

Die Bezeichnung gibt Aufschluss über die Funktionen:

- **Nano**: Kompakte Bauform (45x18mm)
- **33**: Betriebsspannung 3.3V

- **BLE**: Bluetooth Low Energy 5.0
- **Sense**: Integrierte Sensor-Suite

Erweiterungsmöglichkeiten

Für erweiterte Anwendungen wie:

- RGB-Farberkennung
- Objekterkennung
- Gestensteuerung

lässt sich das Board mit dem Arducam OV2640-Kameramodul kombinieren. Diese Kombination eignet sich ideal für Machine-Learning- und KI-Anwendungen – die notwendigen Bibliotheken stehen in der Arduino IDE zur Verfügung.
Folgende Abbildung zeigt den Arduino Nano 33 BLE Sense. 4.1

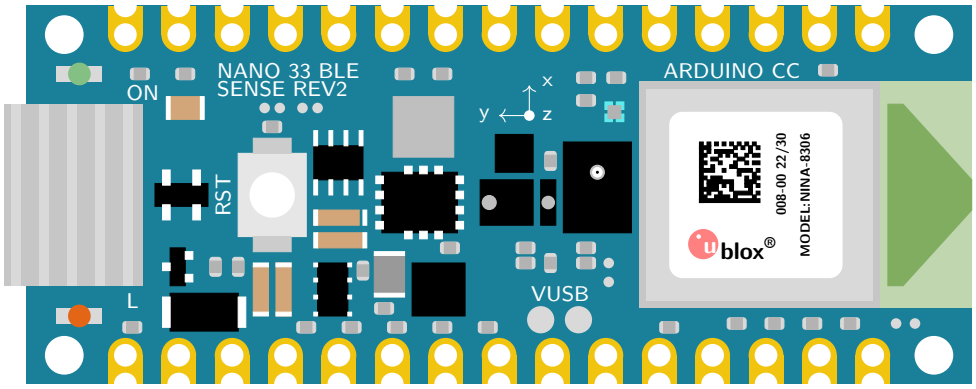


Figure 4.1.: Arduino Nano 33 BLE Sense Rev. 2, see Arduino Store

Technische Spezifikationen des Arduino Nano 33 BLE Sense

Das kompakte und zuverlässige BLE-Board basiert auf dem NINA-B306-Modul für Bluetooth Low Energy (BLE 5.0). Dieses Modul enthält den leistungsstarken Nordic nRF52480-Prozessor mit Cortex-M4F-CPU und ist vollständig kompatibel mit der Arduino IDE (online und offline). [Ard21]

Integrierte Sensoren und Komponenten

- **Funkmodul**:
  - NINA-B306-Modul für Bluetooth 5.0 LE
  - Reichweite bis zu 100m (freies Feld)
  - Datenrate bis zu 2Mbit/s
- **APDS-9960 Sensor**:
  - Integrierter RGB-, Gesten- und Näherungssensor
  - Messbereich: 0-255 für RGB-Werte
  - Gestenerkennung (Links/Rechts/Oben/Unten)
- **LSM9DS1 IMU**:
  - 9-Achsen-Inertialsensor (Beschleunigung, Gyroskop, Magnetometer)

  - Beschleunigungsbereich: ±2/±4/±8/±16 g
  - Gyroskop-Bereich: ±245/±500/±2000 °/s

- **LPS22HB Drucksensor**:
  - Messbereich: 260-1260 hPa
  - Temperaturbereich: -40°C bis +85°C

- **HTS221 Feuchtesensor**:
  - Relative Luftfeuchtigkeit: 0-100
  - Genauigkeit: ±3.5% RH

- **MP34DT05 Mikrofon**:
  - Digitales MEMS-Mikrofon
  - Frequenzbereich: 100Hz-16kHz
  - Signal-Rausch-Verhältnis: 64dB

4.2. Unterschied zwischen dem Arduino Nano 33 BLE Sense Rev1, dem Arduino Nano 33 BLE Sense Rev2 und dem Arduino Nano 33 BLE Sense Lite

4.2.1. Worin unterscheiden sich Rev1 und Rev2?

Die Unterschiede zwischen dem Arduino Nano 33 BLE Sense und der Rev2-Version betreffen folgende Komponenten: IMU, Temperatur- und Feuchtesensor, Mikrofon, Krypto-Chip und Spannungswandler. In der zweiten Revision wurde:

- Die LSM9DS1-IMU durch zwei separate Module ersetzt:
  - BMI270 (Beschleunigungs- und Drehratensensor)
  - BMM150 (Magnetometer)

- Der HTS221-Feuchtesensor durch den präziseren HS3003

- Das Mikrofon MP34DT05 durch den MP34DT06JTR (bei gleichen Spezifikationen)

- Der Spannungswandler MPM3610 durch den MP2322

- Der Krypto-Chip vollständig entfernt

Zusammenfassung der Änderungen

- **IMU**: Ersatz des 9-Achsen-LSM9DS1 durch BMI270 (6-Achsen) + BMM150 (3-Achsen)

- **Umweltsensor**: HTS221 zu HS3003 (höhere Genauigkeit)

- **Mikrofon**: MP34DT05 zu MP34DT06JTR (funktional identisch)

Verbesserte Nutzerfreundlichkeit

- **Stromversorgung**: MPM3610 zu MP2322 (effizienter)
- **VUSB-Lötpad**: Neu auf der Oberseite der Platine
- **Testpunkte**: Für USB, SWDIO und SWCLK hinzugefügt

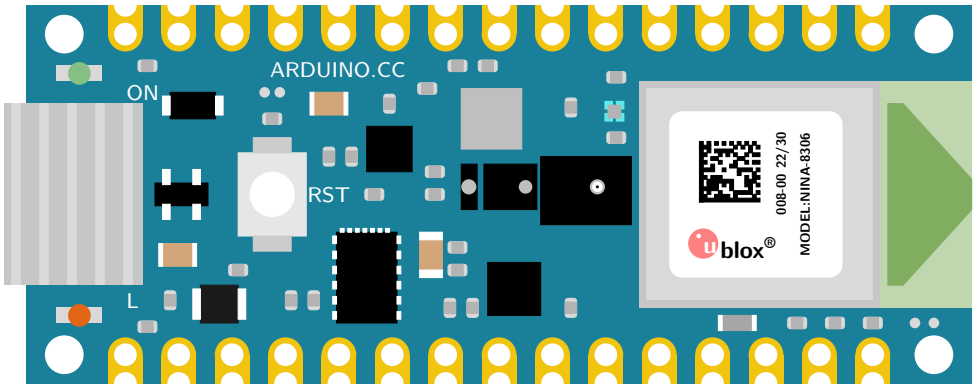


Figure 4.2.: Arduino Nano 33 BLE Sense Rev. 1

Anpassungen in Sketches

Für bestehende Sketches müssen folgende Bibliotheken angepasst werden:

- **IMU**: LSM9DS1 zu ArduinobBMI270bBMM150
- **Umweltsensor**: HTS221 zu Arduino HS300x

Rev 1 TensorFlow Lite Bibliothek: TensorFlow Lite lib

4.2.2. Arduino Nano 33 BLE Sense Lite

Das Tiny Machine Learning Kit enthält ausschließlich die Arduino Nano 33 BLE Sense Lite Version.

Unterschiede zur Rev1

Die Lite-Variante unterscheidet sich nur minimal vom Arduino Nano 33 BLE Sense Rev1:

- **Fehlender Sensor**: Der HTS221 (Feuchtigkeit und Temperatur) wurde entfernt
- **Alternative Temperaturmessung**: Kann über den verbleibenden LPS22HB Drucksensor erfolgen
- **Einschränkung**: Keine Messung der relativen Luftfeuchtigkeit möglich

Technische Hintergründe

- **Entscheidungsgrund**: Lieferkettenprobleme bei der Beschaffung des HTS221 Sensors
- **Kompensation**: Externe Feuchtesensoren können über I²C angeschlossen werden

- **Kostenvorteil**: Geringerer Stückpreis durch reduzierten Sensorumfang

[FD22]

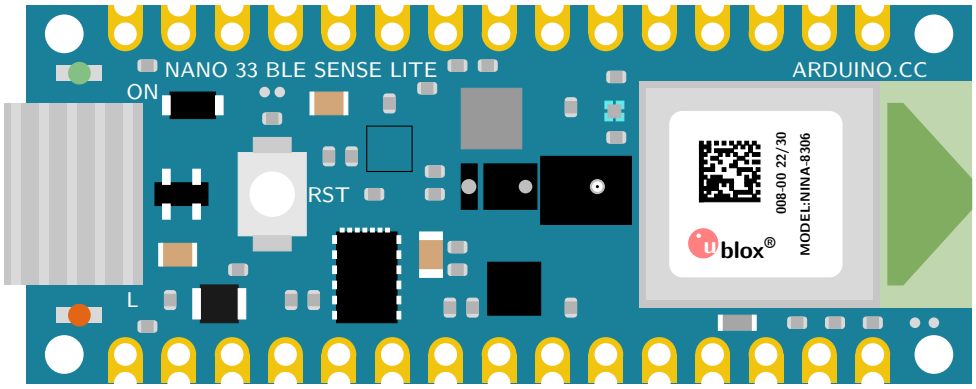


Figure 4.3.: Arduino Nano 33 BLE Sense Lite; Das leere schwarze Quadrat markiert die Position des fehlenden HTS221 Sensors

4.3. Beschreibung der On-Board Sensoren

Der Arduino Nano 33 BLE Sense verfügt über eine Reihe integrierter Sensoren zur Messung analoger und digitaler Umgebungswerte. Mit seinen kompakten Abmessungen von nur 45mm × 18mm eignet er sich ideal für IoT- und KI-Anwendungen, bei denen Platzbeschränkungen eine Rolle spielen. Das Board hat einen geringen Stromverbrauch und arbeitet mit 3,3 V, wodurch es monatelang mit kleinen Batterien betrieben werden kann. Dank der integrierten Sensoren, des niedrigen Energieverbrauchs und der kompakten Bauweise kann dieses Nano-Board vielseitig eingesetzt werden. Der Arduino Nano 33 BLE Sense ist ein völlig neues Board in bekanntem Formfaktor. Detaillierte Informationen zu den Komponenten und Datenblätter der Sensoren finden Sie unter [Ard21].
Die Sensoren im Überblick:

- Der APDS-9960 ist ein digitaler Sensor für Näherung, Umgebungslicht, RGB und Gesten. Er misst Entfernung, Licht, Farbe und Gesten in Boardnähe.
- Der LSM9DS1 ist ein 9-Achsen- Inertial Measurement Unit (IMU) (Beschleunigungsmesser, Gyroskop und Magnetometer), ideal für Wearables.
- Der LPS22HB ist ein barometrischer Drucksensor zur Messung des Umgebungsdrucks, nützlich für einfache Wetterstationen.
- Der HTS221 misst relative Luftfeuchtigkeit und Temperatur für präzise Umgebungsdaten.
- Der MP34DT05 ist ein digitales Mikrofon zur Echtzeiterfassung und Analyse von Geräuschen.
- Der USB-Port ermöglicht die Verbindung mit einem Computer.
- Das Board hat drei LEDs: eine programmierbare RGB-LED, eine orange Status-LED und eine Power-LED.

Abbildung 4.4 zeigt die Anordnung der Sensoren auf dem Board sowie den leistungsstarken nRF52840-Prozessor von Nordic Semiconductors, einen 32-bit ARM® Cortex™-M4 mit 64 MHz Taktfrequenz.

MICROCONTROLLER	nRF52840
OPERATING VOLTAGE	3.3V
INPUT VOLTAGE (LIMIT)	21V
DC CURRENT PER I/O PIN	15 mA
CLOCK SPEED	64MHz
CPU FLASH MEMORY	1MB
SRAM	256KB
LED_BUILTIN	13
IMU (Accelerometer, Gyroscope, Magnetometer)	LSM9DS1
MICROPHONE	MP34DT05
GESTURE, LIGHT, PROXIMITY, COLOUR	APDS9960
BAROMETRIC PRESSURE	LPS22HB
TEMPERATURE, HUMIDITY	HTS221

Table 4.1.: Technische Spezifikation des Arduino Nano 33 BLE Sense [Ard21]

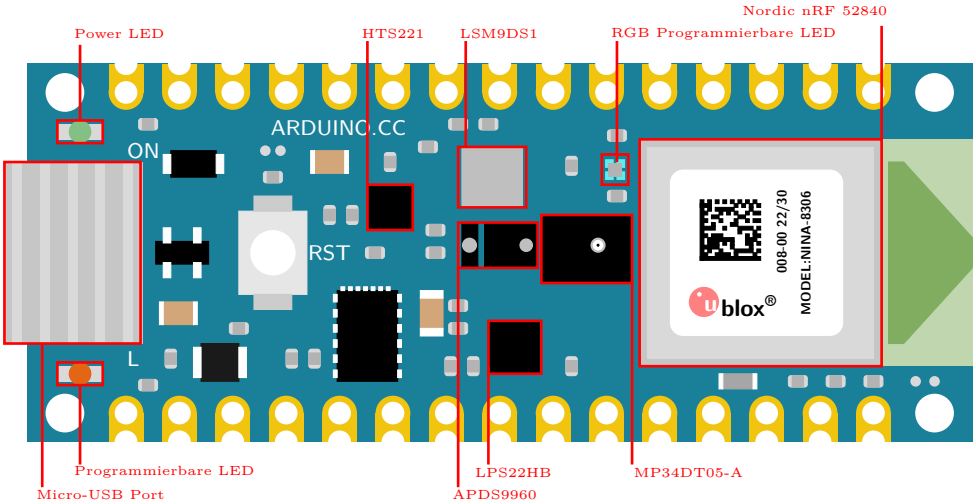


Figure 4.4.

Zu beachten ist die Genauigkeit der Sensorwerte in Abhängigkeit von deren Positionierung. Der Betriebstemperaturbereich liegt zwischen -40°C und +85°C. Auch Luftfeuchtigkeit und Luftdruck sollten berücksichtigt werden.

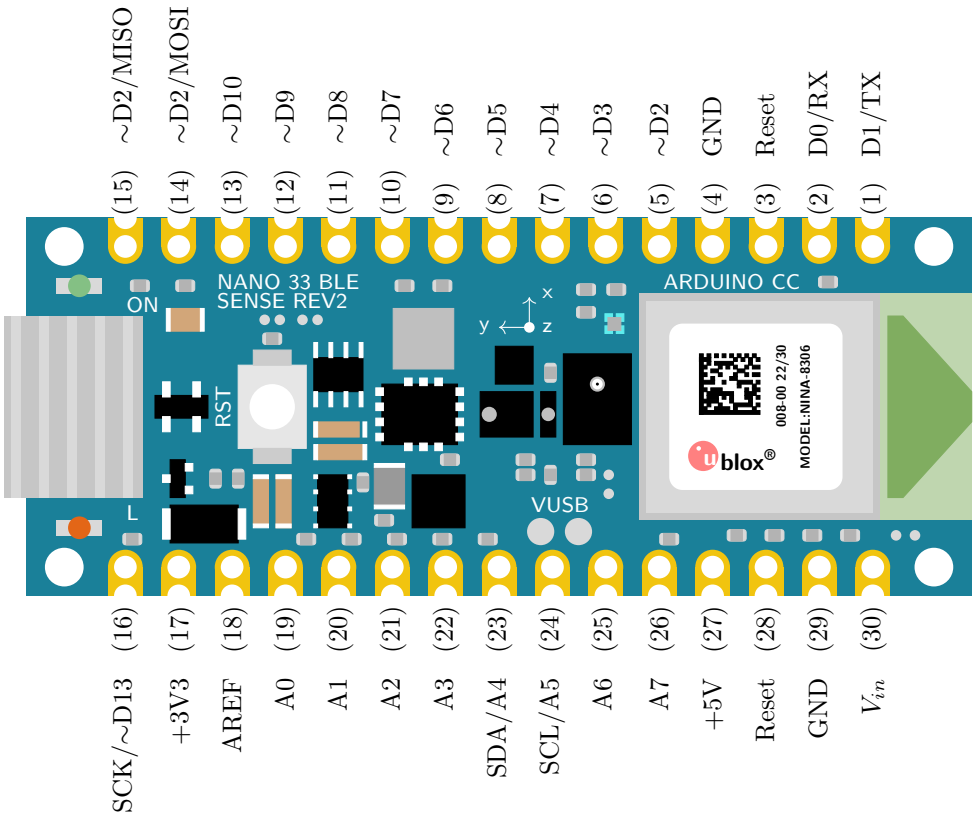


Figure 4.5.: Pinbelegung des Arduino Nano 33 BLE Sense und der Rev 2 Version; Die orange Status-LED ist an Pin D13 angeschlossen, die Power-LED an Pin D1. Die RGB-LED verwendet die Pins D18, D19 und D20.

4.4. Pinbelegung des Arduino Nano 33 BLE

Der Arduino Nano 33 BLE ist eine weiterentwickelte Version des Arduino Nano Boards mit dem leistungsstarken nRF52840 Prozessor. Das Board verfügt über folgende Pin-Konfiguration (siehe auch Pin-Konfiguration):

Digitale Pins 14 digitale I/O-Pins, die nur die Zustände HIGH oder LOW erkennen. Diese Pins können je nach Anforderung als Eingang oder Ausgang konfiguriert werden. Bei 5V liegen sie im HIGH-Zustand, bei 0V im LOW-Zustand.

Analoge Pins 8 analoge Eingänge (A0 bis A7), die im Gegensatz zu digitalen Pins kontinuierliche Werte erfassen können. Der Messbereich liegt zwischen 0-5V.

PWM-Pins Alle digitalen Pins können für Pulsweitenmodulation (PWM) genutzt werden, um analoge Ergebnisse mit digitalen Mitteln zu erzeugen.

SPI-Schnittstelle Unterstützt das Serial Peripheral Interface Protokoll für die Kommunikation mit Peripheriegeräten wie Sensoren. Verwendet die Pins:

- MISO (Master Input Slave Output)
- MOSI (Master Output Slave Input)

I²C-Schnittstelle Zwei-Draht-Protokoll mit den Pins:

- SCL (Serial Clock)
- SDA (Serial Data)

UART-Schnittstelle Serielle Kommunikation über:

- Rx (Empfangs-Pin)
- Tx (Sendepin)

Externe Interrupts Alle digitalen Pins können für externe Interrupts genutzt werden, um laufende Programme bei Bedarf zu unterbrechen.

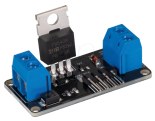
Besondere Pins

- Pin 13: Integrierte LED
- AREF: Referenzspannungseingang

Part II.

Harwarebeschreibung: Magic Faucet Fountain

5. Materialliste Magic Faucet Fountain

Anzahl	Bezeichnung	Link	Preis
1	4DUNIO Wireless Modul HC-05 Bluetooth (4-Pin)	www.reichelt.de - HC-05 Bluetooth-Modul	23,30 €
1	Schaltnetzteil 50 W (5 V / 12 V, 6 A) geschlossen	www.reichelt.de - Schaltnetzteil 5/12V	17,10 €
1	COMET Niedervolt Tauchpumpe 12 V, 600 l/h, 5,5 m	www.conrad.de - COMET Tauchpumpe	14,99 €
1	Blaue LED-Signalleuchte 12 V, 8 mm, 600 mcd	www.reichelt.de - blaue LED-Signalleuchte	6,75 €
1	Grüne LED-Signalleuchte 12 V, 8 mm, 1300 mcd	www.reichelt.de - grüne LED-Signalleuchte	6,75 €
3	N-Kanal MOSFET IRF9540N  (3 × 1,20 €)	www.reichelt.de - IRF9540N MOSFET	3,60 €
1	Miniatur-Kippschalter EIN/AUS 3 A, 125 V	www.reichelt.de - Kippschalter	0,60 €

Gesamtpreis: 85,09 €

Stand: 07.05.2025

6. Ultraschallsensor HC-SR04

6.1. Einleitung

Ein Ultraschallsensor misst berührungslos den Abstand zwischen Sensor und einer Werkstückoberfläche. Aus dieser Distanz kann der aktuelle Abstand zu der Oberfläche berechnet werden. Der Sensor arbeitet nach dem Prinzip der Laufzeitmessung von Ultraschallwellen und bietet eine einfache, kostengünstige und präzise Möglichkeit zur Pegelüberwachung, Warenerkennung und Abstandsmessung.[AG24] [TR18] [HS23]

6.2. Ultraschallsensoren zur Abstandsmessung

Ultraschallsensoren zur Abstandsmessung dienen der Ermittlung von Objektabständen über die Laufzeit von Ultraschallimpulsen und die Schallgeschwindigkeit. Da die Schallgeschwindigkeit temperatur- und luftdruckabhängig ist und zusätzlich durch die Luftzusammensetzung beeinflusst wird, ist bei der Auslegung auf solche Einflussfaktoren Rücksicht zu nehmen. Bei Nichtbeachtung sind Messungenauigkeiten in der Entfernungsmessung zu erwarten. [HS09] Umwelteinflüsse wie Feuchte, Staub und Rauch beeinflussen die Messgenauigkeit nicht. Einige Gase und Flüssigkeiten können die Sensorfunktion jedoch beeinträchtigen oder gar untüchtig machen. Beispiele hierfür sind CO2 und Dämpfe von Lösungsmitteln. [HS+12] Ultraschallsensoren können nach ihrem Sensorprinzip in Tastbetrieb und Schrankenbetrieb unterschieden werden. Der Tastbetrieb basiert dabei auf der Auswertung der Laufzeitmessung zwischen dem Senden und Empfangen des Schalls. Der Schrankenbetrieb dahingegen beruht auf der Kontrolle, ob das gesendete Signal empfangen wurde [HLG19] Der Ultraschallsensor zur Abstandsmessung beruht auf ersterem Prinzip. Im Folgenden soll die Funktionsweise näher erläutert werden. Zunächst aktiviert die Steuerelektronik den Leistungsverstärker, welcher den Schallwandler mit einer elektrischen Leistung und einer Sinusspannung ansteuert. Dadurch arbeitet der Schallwandler als Lautsprecher und sendet einen Ultraschallimpuls, auch Burst genannt, aus. Bis der Schallwandler ausgeschwungen ist, dauert es zwei- bis dreimal so lang wie die Sendezeit. Anschließend leitet die Steuerelektronik den Empfangsbetrieb ein und der Schallwandler dient nun als Mikrofon. Sollte sich in der Schallkeule ein Objekt befinden, wird der Ultraschallimpuls zum Sender zurückreflektiert. Der Schallwandler beginnt zu schwingen und erzeugt eine Sinusschwingung, die auf einen Verstärker übertragen wird. Durch Autokorrelation wird kontrolliert, ob es sich bei dem reflektierten Schall wirklich um das Echo der ausgesandten Ultraschallwellen handelt. [HS09] Eine ganze Bandbreite an Objekten unabhängig von der Oberflächenbescha‘enheit, Farbe, Transparenz [HS+12] und Form [HS09] kann durch den Ultraschall erfasst werden. Der Ultraschall findet seine Anwendung in sehr vielen verschiedenen Bereichen 6.3, näher soll auf ihre Anwendung in den Bereichen der Medizin und Industrie eingegangen werden. In der Medizin gehört der Ultraschall mit zu den wichtigsten und häufigsten genutzten Diagnoseverfahren. Er lässt sich vielfältig einsetzen. So dient es zur Erkennung und Darstellung von Organen, gefüllten Hohlräumen, Gefäßen, Knochen, Sehnen und anderen Objekten im menschlichen Körper. Kennzeichnend für die Anwendung des Ultraschalls in der Medizin ist außerdem sein geringes Gesundheitsrisiko für Patienten. In der Industrie gibt es vielfältige Einsatzbereiche von Ultraschallsensoren. Sie können der Überwachung von Stellplätzen in der Lagerhaltung, der Füllstandbestimmung von Schüttgütern [HS09] oder der Detektion von Objekten und Grenzflächen dienen.

Beispiel für letztere Erwähnung ist die Rückfahrhilfe bei Fahrzeugen. Eine weitere sehr wichtige Anwendung findet sich in der zerstörungsfreien Prüfung von Werkstücken und WerkstoNen wieder. Vorrangig wird dabei das Materialgefüge auf Unebenheiten und Au'älligkeiten überprüft. Solche Untersuchungen werden häufig für sicherheitsrelevante Bauteile wie Eisenbahnräder oder Druckbehälter für Atomanlagen verwendet [MM17].

6.3. Grundlagen zur Verwendung eines Ultraschallsensors

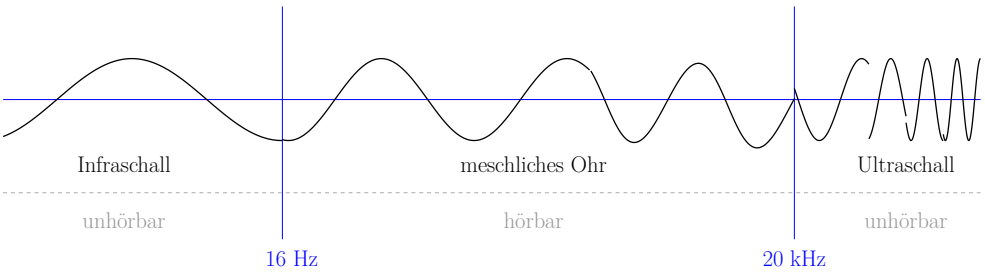
Um einen Ultraschallsensor gezielt und effizient einsetzen zu können, ist ein grundlegendes Verständnis über die Funktionsweise, technischen Eigenschaften und Anwendungsvoraussetzungen abstandsmessender Ultraschallsensoren notwendig. Die folgenden Unterkapitel bieten eine strukturierte Einführung in die physikalischen und technischen Grundlagen des Sensors. Ziel ist es, ein tiefergehendes Verständnis für den Aufbau, die Signalverarbeitung und die praktische Handhabung eines Ultraschallsensors zu vermitteln. [AG24]

6.3.1. Grundlagen Schall

Als Schall bezeichnet man die Ausbreitung von lokalen Druckschwankungen in einem elastischen Medium. Die Schallwelle breitet sich dabei longitudinal aus, die Auslenkung der Moleküle erfolgt also in Ausbreitungsrichtung [TM24]. Die Einteilung der Schallbereiche erfolgt auf Grundlage der Frequenz, siehe außerdem 6.1:

- *Infraschall:* bis 16 Hz,
- *Hörbarer Schall:* von 16 Hz bis 20 kHz,
- *Ultraschall:* von 20 kHz bis 1 GHz,
- *Hyperschall:* über 1 GHz [HS23].

Figure 6.1.: Frequenz der verschiedenen Schall-Arten



6.3.2. Schallgeschwindigkeit und Wellenlänge

Die Geschwindigkeit, mit der sich Schall in Luft ausbreitet, ist von der Lufttemperatur T und der Luftfeuchtigkeit φ abhängig. Da die Luftfeuchtigkeit einen erheblich geringeren Einfluss auf den Wert hat

$$\Delta v_{Schall,\varphi} \approx 1,2\,\frac{\mathrm{m}}{\mathrm{s}};\ 0 < \varphi < 1,\ T_C = 20\,^{\circ}\mathrm{C}, \tag{6.1}$$

[HS23] lässt sich die Schallgeschwindigkeit mit ausreichender Genauigkeit mit der Formel 6.2

$$v_{Schall} = \sqrt{\kappa R T} \tag{6.2}$$

und der Annahme $\varphi = 0$ berechnen [TM24]. Dabei ist $\kappa = 1{,}4$ der Adiabatenexponent von Luft, $R = 287\ \frac{\mathrm{J}}{\mathrm{kg\ K}}$ die spezifische Gaskonstante von Luft und T die absolute Temperatur in Kelvin [TM24; Wil22]. Für eine angenommene Temperatur von $T_C = 20\,°\mathrm{C}$ ergibt sich damit eine Geschwindigkeit von $v_{Schall} = 343,2\ \frac{\mathrm{m}}{\mathrm{s}}$. In Abb. 6.2 ist die Schallgeschwindigkeit in Luft im Temperaturbereich von $T_C = -10\,°\mathrm{C}$ bis $60\,°\mathrm{C}$ dargestellt.

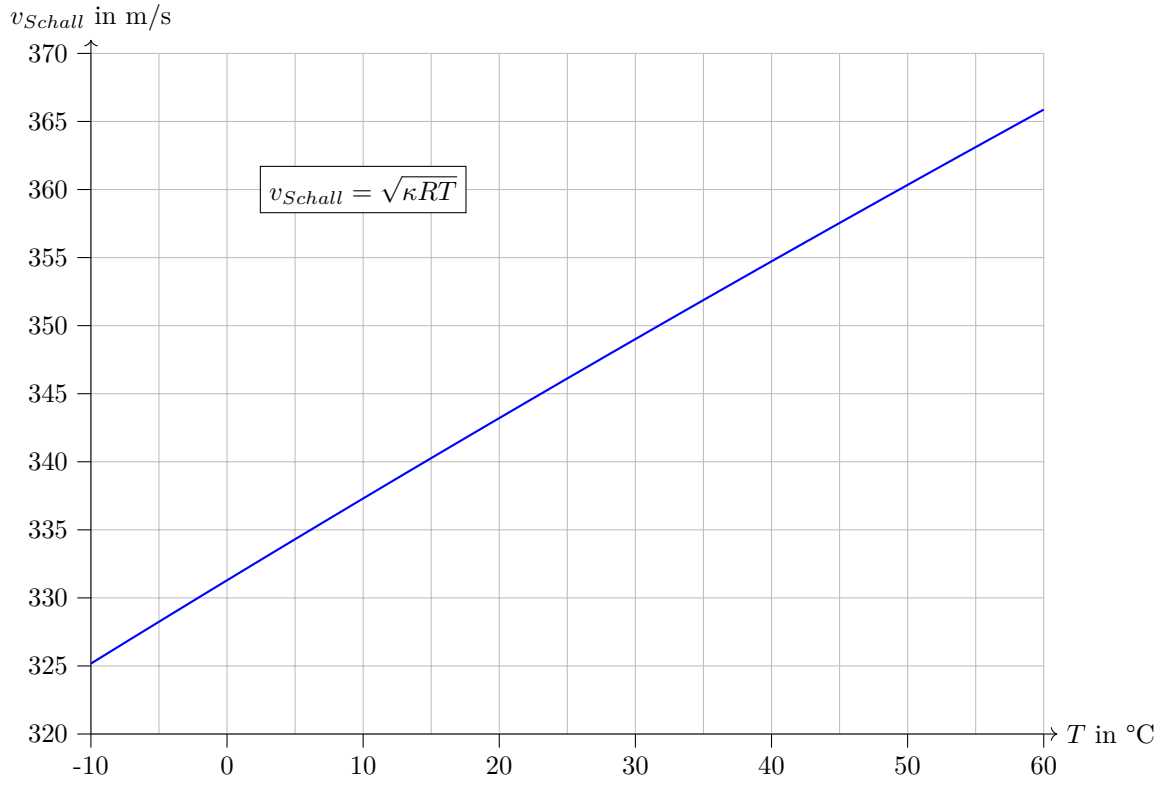


Figure 6.2.: Ausbreitungsgeschwindigkeit von Schall in Luft

Über den Formelzusammenhang 6.3

$$v_{Schall} = \lambda f \tag{6.3}$$

lässt sich bei gegebener Frequenz f und mit der berechneten Schallgeschwindigkeit v_{Schall} die Wellenlänge des Schalls berechnen [Wil22]. Diese beträgt bei einer angenommenen Frequenz von $f = 40\mathrm{kHz}$ und der in Abschnitt 6.3.2 berechneten Schallgeschwindigkeit $\lambda = 8{,}58 \cdot 10^{-3}$ m.

6.3.3. Weg- und Abstandsmessung mit Ultraschall

Um eine Entfernung oder einen Abstand eines Objekts zu dem Sensor zu ermitteln, wird der Sensor im sog. *Tastbetrieb mit Echo-Laufzeit-Messung*, wie in Abb. 6.3 zu sehen, verwendet. Dabei sendet der Sensor zunächst ein Schallpuls in Richtung des zu messenden Objekts aus (Lautsprecher), welche zum Teil von diesem reflektiert wird und somit zurück zum Sensor gelangt (Echo). Dieses Echo kann der Sensor detektieren (Mikrofon) und über die gemessene Laufzeit des Echos sowie die Schallgeschwindigkeit im dem den Sensor umgebenden Medium die Distanz berechnen [HS23].

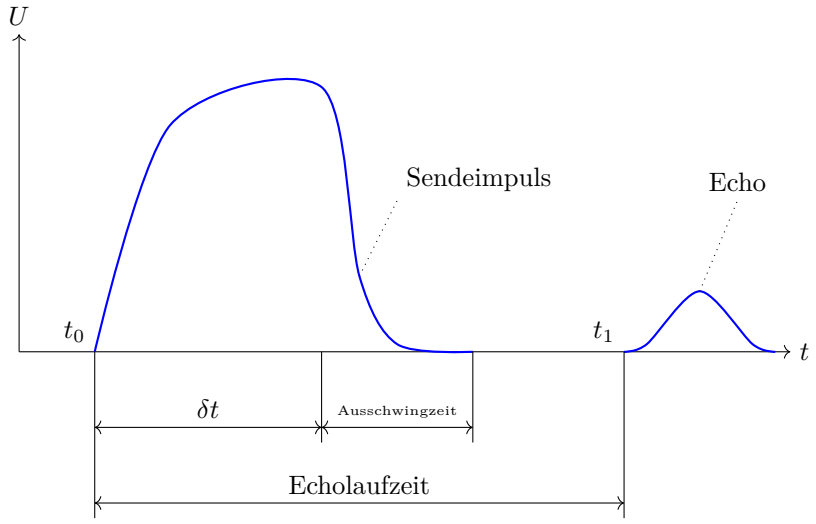
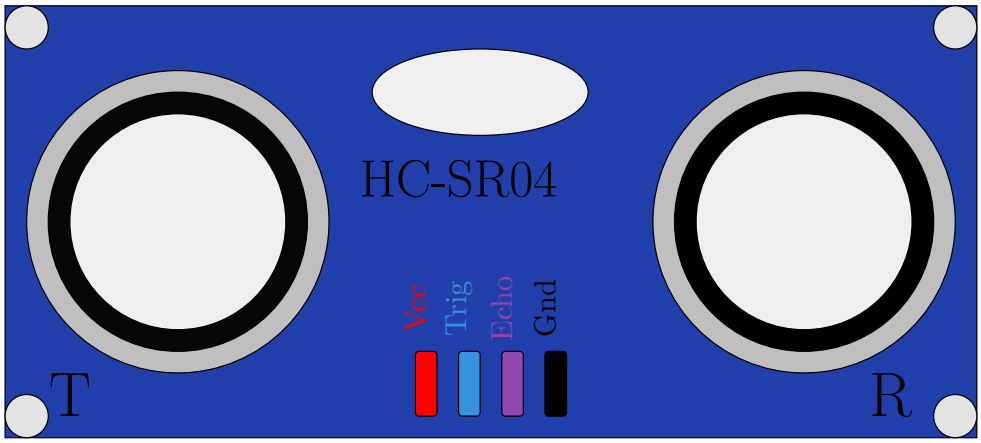


Figure 6.3.: Spannungsverlauf Schallwandler bei einer Echo-Laufzeit-Messung, nach [HS23]

6.4. Ultraschall-Abstandsensor HC-SR04

Der Ultraschallabstandssensor in Abbildung 6.4 enthält den Sensor HC-SR04. Er ist als zweiköpfiger Sensor ausgeführt, wodurch er über je einen Lautsprecher und ein Mikrofon verfügt und ist zur Erfassung von Abständen in einem Bereich von 3 bis 400 cm spezifiziert. Dafür nutzt er eine Schallfrequenz von 40 kHz. Er ist als zweiköpfiger Sensor ausgeführt, wodurch er über je einen Lautsprecher und ein Mikrofon verfügt. Die Abmessungen des Sensors betragen in der Breite 43 mm, in der Höhe 15 mm und in der Tiefe 20 mm. Die Auflösung des Sensors beträgt ±1 mm. Angeschlossen wird der Abstandssensor über den Voltage at the Common Collector (VCC)-Pin, Trig/Serial Clock (SCL)-Pin, Echo/Serial Data (SDA)-Pin und dem Ground (GND)-Pin. Der Sensor kann über die drei Schnittstellen General Purpose Input Output (GPIO), Universal Asynchronous Receiver Transmitter (UART) und Inter-Integrated Circuit (I^2C) verbunden werden [Sim].

Figure 6.4.: Ultraschallsensor mit Pin-Belegung



6.5. Spezifikation

Der Sensor HC-SR04 ist zur Erfassung von Abständen in einem Bereich von 3 bis 400 cm mit einer Auflösung von ±1mm spezifiziert. Dafür nutzt er eine Schallfrequenz von 40 kHz. Die Abmessungen des Sensors betragen in der Breite 43 mm, in der Höhe 15 mm und in der Tiefe 20 mm.
Angeschlossen wird der Abstandssensor über den SCL-Pin für das Triggersignal, den SDA-Pin fürs Echo, den VCC-Pin zur Stromversorgung und dem GND-Pin.
Der Sensor kann über die drei Schnittstellen GPIO, UART und I^2C verbunden werden [Sim].
Bei den folgenden Beispielprogrammen wird der Arduino Nano 33 BLE Sense verwendet. In den Beispielen werden die Pins D6 und D9 für die Datenübertragung verwendet. In der Tabelle 6.1 ist dies für die Grove-Schnittstelle festgehalten und zudem auch in der Abbildung 6.4 zu sehen.

Table 6.1.: Pinbelegung für den Ultraschallabstandssensor [Sim]

Arduino Pin	Ultraschall Abstandssensor
D9	Trig
D6	Echo
3V3	VCC
GND	GND

6.5.1. Anwendungen

Ultraschallsensoren finden in der Praxis Anwendung in verschiedenen Bereichen, die auf die Ausbreitung des Schalls über unterschiedliche Medien basieren [HS23]. Es lassen sich drei Hauptkategorien unterscheiden: In der ersten Kategorie breitet sich der Schall über Luft aus. Typische Anwendungen sind hier Füllstandmessungen, Durchmessererfassungen bei Auf- und Abwickelsteuerungen sowie Kollisionsschutz und Objekterkennung [HS23]. Die zweite Kategorie umfasst Sensoren, bei denen sich der Schall über Flüssigkeiten ausbreitet. Diese finden vor allem im maritimen Bereich Verwendung, beispielsweise in Sonarsystemen zum Orten von Unterseebooten oder Fischschwärmen sowie in Echoloten zur Kartografie des Meeresbodens [HEG21]. Die dritte Kategorie umfasst Körperschallsensoren, die in der Industrie zur zerstörungsfreien Materialprüfung sowie in der Medizin zur Ultraschalldiagnostik verwendet werden. [HEG21]

6.6. Bibliothek

6.6.1. Beschreibung

Zur Ansteuerung des Ultraschallsensors HC-SR04 wird eine Arduino-kompatible Bibliothek `NewPing` verwendet, die die Initialisierung des Sensors sowie das Senden und Empfangen von Ultraschallsignalen abstrahiert. Diese Bibliothek kapselt die Funktionen zum Triggern, zur Laufzeitmessung und zur Berechnung der Distanz [AG24].

6.6.2. Installation

Die Installation der Bibliothek `NewPing`, welche die Funktionalität des Ultraschallsensors HC-SR04 unter Arduino abstrahiert, erfolgt über den Bibliotheksverwalter der Arduino IDE. Diese bietet eine benutzerfreundliche Oberfläche zur Integration externer Bibliotheken. [AG24]
Schrittweise Anleitung:

1. Öffne die Arduino IDE (Version 1.8.x oder neuer empfohlen).

2. Navigiere zu: Sketch > Bibliothek einbinden > Bibliotheken verwalten...
3. Im Suchfeld oben rechts den Begriff `NewPing` eingeben
4. Wähle den Eintrag `NewPing` by Tim Eckel aus (Version 1.9.7 oder neuer empfohlen).
5. Klicke auf „Installieren “.

6.6.3. Funktionen

Die NewPing-Bibliothek bietet eine Vielzahl von Funktionen, die die Nutzung von Ultraschallsensoren erleichtern und erweitern. Die wichtigsten Funktionen sind: [AG24]

`NewPing()`: Initialisierung eines Sensors durch Angabe der Pins für Trigger- und Echo-Signale sowie der maximalen Messdistanz.
`ping()`: Senden eines Ultraschallimpulses und Rückgabe der gemessenen Laufzeit in Mikrosekunden.
`getDistance()`: Direkte Ausgabe der gemessenen Entfernung in Zentimetern basierend auf der Standard-Schallgeschwindigkeit in Luft.
`pingMedian()`: Durchführung mehrerer Messungen und Rückgabe des Medianwertes zur Reduktion von Ausreißern.

Diese Funktionen ermöglichen eine einfache und präzise Nutzung von Ultraschallsensoren in verschiedenen Anwendungen, von der Einzelmessung bis hin zur Durchführung mehrerer Messungen zur Reduzierung von Ausreißern.

6.7. Kalibrierung

Damit der HC-SR04 präzise misst, muss er kalibriert werden. Die wichtigste Einflussgröße ist die Schallgeschwindigkeit, da sie von der Umgebungstemperatur abhängt. Eine Kalibrierung erfolgt durch den Vergleich mit einer bekannten Referenzdistanz. Ergibt sich eine systematische Abweichung, kann diese durch einen festen Korrekturwert (Offset) oder durch Anpassung der Schallgeschwindigkeit ausgeglichen werden. [TR14] Der Ultraschallabstandssensor wird für die Verwendung bei der Temperatur von 20 °C ausgelegt. Die berechnete Schallgeschwindigkeit $v_{Schall} = 343,2\,\frac{\mathrm{m}}{\mathrm{s}}$ bei 20 °C wird für die Umrechnung der Schalldauer in die Distanz zur Reflexionsfläche verwendet.

6.8. Simple Code

Zur Überprüfung der grundlegenden Funktionalität des Ultraschallsensors wurde ein Simple Code 6.1 verwendet. Das Programm dient zum einfachen Test mithilfe der NewPing-Bibliothek. Es misst die Laufzeit (Echozeit) des Schallsignals . Die Werte werden anschließend über den Seriellen Monitor ausgegeben.

6.9. Einfache Anwendung

In der Industrie ist es oft wichtig zu wissen, wann ein bestimmter Pegel einer Flüssigkeit unterschritten wird. Dafür können Ultraschallsensoren eingesetzt werden. Sie messen kontinuierlich den Pegel und können dem Programm so permanent einen Messwert bereitstellen. Wird ein bestimmter Messwert über- oder unterschritten, kann das Programm eine Warnmeldung erzeugen. Dieses Arduino-Programm 6.2 nutzt einen

Listing 6.1.: Einfacher Code zum Testen des Ultraschallsensors

```
/**
 * @file laufzeitMessung.ino
 * @brief Simple test program for measuring the echo time of an HC-SR04 ultrasonic sensor using the NewPing library.
 *
 * This program initializes the ultrasonic sensor and performs a basic runtime measurement (ping)
 * to determine the echo time in microseconds. The result is printed to the Serial Monitor.
 */

#include <NewPing.h>

// Define pins and maximum distance
#define TRIGGER_PIN  9    ///< Pin connected to the trigger pin of the HC-SR04
#define ECHO_PIN     10   ///< Pin connected to the echo pin of the HC-SR04
#define MAX_DISTANCE 200  ///< Maximum distance to measure (in cm)

/// Create a NewPing object named `sonar`
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

/**
 * @brief Arduino setup function.
 *
 * Initializes serial communication for output.
 */
void setup() {
  Serial.begin(9600);
  Serial.println("Ultrasonic runtime measurement started...");
}

/**
 * @brief Arduino main loop function.
 *
 * Measures the echo time using the ping() function and prints it in microseconds.
 */
void loop() {
  delay(1000); // Wait 1 second between measurements

  // Measure echo time in microseconds
  unsigned int echoTime = sonar.ping();

  // Print result
  Serial.print("Echo time: ");
  Serial.print(echoTime);
  Serial.println("mikro-sekunden");
}
```

../../Code/Arduino/UltraSonicHCSR04/laufzeitMessung/laufzeitMessung.ino

HC-SR04 Ultraschallsensor, um die Entfernung zu einem Objekt kontinuierlich zu überwachen. Es erkennt, ob eine kritische Schwelle (25 cm) unterschritten wurde und gibt in diesem Fall eine Warnmeldung im Seriellen Monitor aus.

Listing 6.2.: Einfache Anwendung als Schwellenwert-Melder

```
/**
 * @file schwellenwertMeldung.ino
 * @brief Simple application using an HC-SR04 ultrasonic sensor to monitor a distance threshold.
 *
 * This program continuously measures the distance using the HC-SR04 sensor and triggers a warning
 * when a critical threshold of 25 cm is reached or undershot. The distance is measured using
 * pingMedian() for stable values, and getDistance() is also demonstrated for direct measurement.
 */

#include <NewPing.h>

#define TRIGGER_PIN     9    ///< Pin connected to the trigger pin of the HC-SR04
#define ECHO_PIN        10   ///< Pin connected to the echo pin of the HC-SR04
#define MAX_DISTANCE    100  ///< Maximum distance to measure (in cm)

#define TARGET_DISTANCE     30  ///< Reference distance (in cm)
#define CRITICAL_THRESHOLD  25  ///< Critical threshold for warning (in cm)

/// Create a NewPing object named `sonar`
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

/**
 * @brief Arduino setup function.
 *
 * Initializes serial communication and outputs startup message.
 */
void setup() {
  Serial.begin(9600);
  Serial.println("Distance threshold monitoring started...");
}

/**
 * @brief Arduino main loop function.
 *
 * Continuously measures the distance using pingMedian().
 * If the measured distance is less than or equal to the critical threshold,
 * a warning message is printed.
 */
void loop() {
  delay(500); // Measurement interval

  // Use pingMedian() for stable distance reading (average of multiple pings)
  unsigned int distance = sonar.pingMedian();

  // Print measured distance
  Serial.print("Measured distance: ");
  Serial.print(distance);
  Serial.println(" cm");

  // Check if distance is at or below critical threshold
  if (distance > 0 && distance <= CRITICAL_THRESHOLD) {
    Serial.println("WARNING: Critical distance reached!");
  }

  // Optional: demonstrate use of getDistance() (less accurate, uses default single ping)
  // unsigned int quickDistance = sonar.getDistance();
  // Serial.print("Quick distance (getDistance): ");
  // Serial.print(quickDistance);
  // Serial.println(" cm");
}
```

../../Code/Arduino/UltraSonicHCSR04/schwellenwertMeldung/schwellenwertMeldung.ino

6.10. Tests

6.10.1. Einfacher Funktionstest

Beim einfachen Funktionstest wird überprüft, ob der Sensor korrekt anspricht und brauchbare Messwerte liefert. Dazu wird der HC-SR04 an einen Mikrocontroller (z. B. Arduino Nano 33 BLE Sense Lite) angeschlossen und ein einfacher Sketch 6.3 aufgespielt, welcher die gemessene Entfernung über die serielle Schnittstelle ausgibt. [HS23] Ein Testobjekt wird in einem bekannten Abstand, typischerweise 20 cm, orthogonal vor den Sensor gehalten. Die gemessene Entfernung wird mit einem Maßband überprüft. Stimmen beide Werte überein, ist die Grundfunktion des Sensors sichergestellt. Dieser Test eignet sich insbesondere zur ersten Inbetriebnahme oder zur Überprüfung von Lötverbindungen, Spannungsversorgung und Signalverarbeitung.

Listing 6.3.: Code für die Messung einer Strecke

```
/**
 * @file entfernungMessung.ino
 * @brief Simple program to measure and display distance in centimeters using the HC-SR04 ultrasonic sensor.
 *
 * This program initializes the HC-SR04 sensor and continuously measures the distance to an object.
 * The distance is displayed in centimeters on the Serial Monitor using the NewPing library.
 */

#include <NewPing.h>

// Define pins and maximum measurement distance
#define TRIGGER_PIN    9      ///< Pin connected to the trigger pin of the HC-SR04
#define ECHO_PIN       10     ///< Pin connected to the echo pin of the HC-SR04
#define MAX_DISTANCE   200    ///< Maximum distance to measure (in cm)

/// Create a NewPing object for distance measurement
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

/**
 * @brief Arduino setup function.
 *
 * Initializes serial communication and prints startup message.
 */
void setup() {
  Serial.begin(9600);
  Serial.println("Simple distance measurement started...");
}

/**
 * @brief Arduino main loop function.
 *
 * Measures the distance using getDistance() and prints it in centimeters.
 */
void loop() {
  delay(1000); // Wait 1 second between measurements

  // Measure distance in cm (automatically calculated by NewPing)
  unsigned int distance = sonar.getDistance();

  // Print result
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
}
```

../../Code/Arduino/UltraSonicHCSR04/entfernungMessung/entfernungMessung.ino

6.10.2. Test aller Funktionen

Ein umfassender Funktionstest geht über die reine Abstandsmessung hinaus. Hierbei werden systematisch verschiedene Einflüsse auf die Messgenauigkeit überprüft: [HS23]

- **Oberflächenbeschaffenheit:** Glatte, harte Oberflächen liefern bessere Reflexionen als weiche oder absorbierende Materialien wie Schaumstoff oder Stoff

- **Ausrichtungswinkel:** Objekte, die schräg zur Sensorachse stehen, reflektieren das Signal schlechter oder gar nicht. Der Test prüft, in welchem Winkelbereich

zuverlässige Ergebnisse erzielt werden.

- **Entfernungsbereich:** Es wird getestet, ob der Sensor sowohl im Nahbereich (unter 10 cm) als auch im Fernbereich (bis 400 cm) verlässlich misst.

- **Umwelteinflüsse:** Staub, leichte Luftströmungen oder Temperaturveränderungen werden simuliert, um die Stabilität des Sensors unter realistischen Bedingungen zu bewerten.

- **Reproduzierbarkeit:** Mehrfache Messungen unter gleichen Bedingungen sollten ähnliche Ergebnisse liefern. Eine Standardabweichung der Messwerte kann berechnet werden, um die Präzision zu bewerten.

6.10.3. Lösungsansätze

Die Genauigkeit von Ultraschall-Abstandsmessungen hängt unter anderem von der Schallgeschwindigkeit ab, welche wiederum temperaturabhängig ist. Wird die Geschwindigkeit im Auswertungsprozess als konstanter Wert angenommen, kann es bei Abweichungen von der tatsächlichen Umgebungstemperatur zu Messfehlern kommen. Eine Möglichkeit zur Korrektur besteht darin, die aktuelle Temperatur zu erfassen und die Schallgeschwindigkeit entsprechend anzupassen. In vielen Systemen wird hierfür ein interner Temperatursensor verwendet. Liegt dieser jedoch direkt auf der Platine und ist in einem Gehäuse untergebracht, kann es durch Eigenwärme und Isolation zu einer zeitverzögerten oder verfälschten Temperaturmessung kommen. Die Verwendung eines separaten Temperatursensors außerhalb des Gehäuses wäre in solchen Fällen eine mögliche Lösung, ist jedoch mit zusätzlichem Aufwand verbunden. Für das vorliegende Projekt, bei dem ein visueller Effekt eines scheinbar schwebenden Wasserhahns erzeugt werden soll, ist eine sehr hohe Messgenauigkeit nicht zwingend erforderlich. Der Wasserstand im Behälter wird zusätzlich durch Bohrungen geregelt, die ein Überlaufen verhindern. Die Ultraschallmessung dient primär der groben Überwachung des Füllstandes, beispielsweise zur rechtzeitigen Nachfüllung. Geringe Messabweichungen durch temperaturbedingte Schwankungen der Schallgeschwindigkeit sind daher tolerierbar und haben keinen wesentlichen Einfluss auf die Funktionalität der Installation.

6.11. Weiterführende Anwendungen

Ultraschallsensoren wie der HC-SR04 finden in folgenden Gebieten ihre Anwendung:

- **Robotik:** Hinderniserkennung und Kollisionsvermeidung bei autonomen Systemen [TR14]

- **Industrieautomation:** In Tanks und Silos, auch bei Staub oder Dampf [HS23].

- **Fahrzeugtechnik:** Einparkhilfen und Abstandswarnsysteme [HS23].

- **Medizin:** Kontaktlose Abstandsmessung in sterilen Umgebungen, z. B. bei Desinfektionsstationen [HS23]

Die Stärken des Sensors liegen in der einfachen Integration, dem günstigen Preis und der Robustheit gegenüber Lichtverhältnissen oder Verschmutzung. Die genannten Eigenschaften werden auch in der Fachliteratur als Vorteile von Ultraschallsensoren gegenüber optischen Verfahren hervorgehoben [HS23]

6.12. Further Readings

Tränkler, H.-R.:mSensortechnik – Handbuch für Praxis und Wissenschaft. 2. Aufl., Springer Vieweg, 2018 [TR18]

- Ein schöner Einstieg und Überblick über die Sensortechnik und ihre Anwendung heutzutage.

Hering, E.: Sensoren in Wissenschaft und Technik – Funktionsweise und Einsatzgebiete. 3. Aufl., Springer Vieweg, 2023 [HS23]

- Ein tiefer Einblick in die Sensortechnik, besonders schöne Abbildungen und Erklärungen. Von Grund auf an eine Einführung für Ultraschallsensoren

Arduino AG: NewPing – Library Documentation, Arduino.cc, abgerufen 2025. [AG24]

- Auf der Arduino Website kann man sich viele Extra-Informationen zum Arduino und weiterführende Anwendungen, Beispiele und Projekte anschauen.

7. Comet Tauchpumpe ELEGANT 12V

7.1. Einführung

Pumpen sind Einrichtungen zur Förderung von flüssigen Medien von einem niedrigeren auf ein höheres statisches Druckniveau. Dieser Vorgang wird über verschiedene Wege erreicht. Ein höheres statisches Druckniveau kann durch folgende Verfahren erreicht werden:

- das Einwirken einer Druckkraft,
- die Übertragung mechanischer Arbeit,
- den Impulsaustausch,
- das Zumischen von Druckluft zum Förderwasser,
- die Stoßwirkung einer in ihrer Bewegung gehemmten Wassersäule oder
- durch Ausnutzung der Zähigkeit des Mediums unter Verwendung eines Förder-gewindes

All die verschiedenen Arbeitsweisen beeinflussen die Pumpenbauart. Die häufigsten vorkommenden Pumpen sind die Verdränger- und Kreiselpumpe. Im Folgenden wird näher auf die Kreiselpumpe eingegangen. Kreispumpen gibt es in ein- oder mehrstufigen Ausführungen. Ihre Arbeitsweise ist gleich, sie unterscheiden sich lediglich in ihrem Aufbau.[Sch13] Pumpen sind oft Komponenten von komplexen Prozessen, deren Ausfall erhebliche Auswirkungen verursachen kann. Die Anforderung an maximale Betriebssicherheit ist daher von besonderer Bedeutung.[GG20]

7.1.1. Arbeitsweise Kreiselpumpen

Die Kreiselpumpe verfügt über ein rotierendes Laufrad mit Schaufeln. Dieses Rad über-trägt mechanische Arbeit auf die in den Laufradkanälen befindliche Flüssigkeit, die da-bei durch Fliehkräfte aus dem Laufrad verdrängt wird. Durch den Vorgang kommt es zur Druckerhöhung und Geschwindigkeitszunahme des Mediums. Die gleichzeitige Zunahme der Absolutgeschwindigkeit des Fördermittels stellt eine unerwünschte Strömungsbegleiterscheinung dar, da in der Pumpe vorrangig eine Druckerhöhung erzielt werden soll. Daher muss diese anschließend in Druckenergie umgewandelt werden. Die Umwandlung wird durch das um das Laufrad befindliche ringförmige Leitrad realisiert. Zusammen bilden Lauf- und Leitrad eine Stufe (daher einstufige Kreiselpumpe). Die kontinuierliche Aufrechterhaltung der Strömung erfolgt dadurch, dass durch die aus dem drehenden Laufrad verdrängte Flüssigkeit eine Saugwirkung erzeugt und das gleiche Volumen an Flüssigkeit über Saugstutzen wieder in die Pumpe eintritt. Die Wahl einer Kreiselpumpe erfolgt nach Förderhöhe und Druckhöhe. So kann nach der Förderhöhe in Nieder, - Mittel – und Hochdruckpumpen unterteilt werden. Im Hin-blick auf die Druckhöhe, wählt man bei großen Druckhöhen mehrstufige Kreiselpumpen. Sie sind im Aufbau wie die einstufigen Kreiselpumpen mit dem einzigen Unter-schied, dass mehrere Stufen (gleichartige Lauf- und Leiträder) hintereinandergeschaltet sind. Dabei wird die Anzahl der Stufen durch die Förderhöhe, Drehzahl der Pumpe und die Größe des Volumenstromes bestimmt. Bei geringen Förderhöhen und großen Volumenströmen entspricht die Laufradform, die dem Francisrade bei Wasserturbinen. Die Schaufeln des Laufrades sind bei dieser Bauart räumlich gekrümmt bzw. verwunden. Eine Erhöhung

des Volumenstroms bei gering bleibender Förderhöhe erfordert eine mehrströmige Ausführung der Pumpe. Bei dieser Bauart sind mehrere Laufräder parallelgeschaltet. Verwirklicht wird das Ganze durch zwei Einzelräder, welche zu einem Radpaar mit doppelseitigem Wassereintritt vereinigt werden. Beispiele für mehrströmige Pumpen sind Kühlwasserpumpen für Kondensationsanlagen. Eine weitere Anwendung bei gleichbleibenden Bedingungen (Vergrößerung Volumenstrom, klein bleibende Förderhöhe) findet die Pumpe halbaxialer Bauart. Diese Bauart zeichnet sich dadurch aus, dass die axial eintretende Flüssigkeit nur noch eine geringe Umlenkung nach der radialen Richtung erfährt. Sie ersetzt die mehrstufige und bildet den Übergang von der Radial- zur Axialpumpe. Reine Axialpumpen eignen sich besonders zur Überwindung großer Volumenströme bei geringen Strecken.[Sch13]

7.2. Charakteristische Größen von Pumpen

Pumpen werden durch charakteristische Größen beschrieben. Im Folgenden sollen die Hauptparameter näher erläutert werden.

7.2.1. Volumenstrom

Der Volumenstrom oder Durchsatz gibt an, wie viel Raum ein Medium pro Zeiteinheit durchströmt. Er wird definiert durch:

$$\dot{V} = \frac{\dot{m}}{\rho} \tag{7.1}$$

Die Einheit ist beispielsweise m^3/h, m^3/s, l/s oder l/min.
Der Volumenstrom setzt sich aus dem effektiv geförderten Volumenstrom und dem Leckvolumenstrom zusammen. Daraus ergibt sich der äußere und innere Dichtheitsgrad:

$$\lambda = \frac{\dot{V}_s}{\dot{V}_i} \tag{7.2}$$

Der Dichtheitsgrad ist dimensionslos und liegt typischerweise zwischen 0,92 und 0,98.

7.2.2. Förderhöhe

Die Förderhöhe (auch Nutzförderhöhe) ergibt sich aus der Summe der Saughöhe H_s und der Druckhöhe H_D:

$$H = H_s + H_D \tag{7.3}$$

Für eine genauere Beschreibung der Förderhöhe werden zusätzlich Druckverluste und Geschwindigkeitsänderungen berücksichtigt. Die vollständige Formel lautet:

$$H = h_{\text{geo}} + \frac{p_D - p_S}{g \cdot \rho} + \frac{c_D^2 - c_S^2}{2g} + H_V \tag{7.4}$$

Diese setzt sich zusammen aus der geodätischen Höhe, den statischen Drücken im Druck- und Saugbehälter, den mittleren Strömungsgeschwindigkeiten im Aus- und Eingangsquerschnitt der Pumpe und den Druckverlusten.

7.2.3. Nutzleistung

Die Nutzleistung ergibt sich aus dem Massenstrom und der spezifischen Nutzarbeit der Pumpe:

$$P_N = g \cdot \rho \cdot \dot{V} \cdot H \tag{7.5}$$

7.2.4. Pumpenwirkungsgrad

Der Pumpenwirkungsgrad berechnet sich als Verhältnis von Nutzleistung zur Kupplungsleistung:

$$\eta_P = \frac{P_N}{P_K} \tag{7.6}$$

Ist der hydraulische, volumetrische, Radreibungs- und mechanische Wirkungsgrad bekannt, so ergibt sich der gesamte Pumpenwirkungsgrad als Produkt:

$$\eta_P = \eta_h \cdot \eta_{\text{vol}} \cdot \eta_R \cdot \eta_m \tag{7.7}$$

Für Kreiselpumpen gelten typischerweise folgende Wirkungsgradbereiche:

- Hydraulischer Wirkungsgrad: $\eta_h = 0{,}82 - 0{,}95$
- Mechanischer Wirkungsgrad: $\eta_m = 0{,}95 - 0{,}98$
- Motorwirkungsgrad: $\eta_M = 0{,}82 - 0{,}95$
- Kupplungswirkungsgrad: $\eta_K = 0{,}82 - 0{,}95$

7.2.5. Kupplungsleistung

Die Kupplungsleistung ist die mechanische Leistung, die zum Antrieb der Pumpe erforderlich ist. Sie setzt sich aus allen Leistungsverlusten zusammen:

$$P_K = P_N + P_h + P_{\text{vol}} + P_R + P_m \tag{7.8}$$

Dazu gehören hydraulische und mechanische Verluste, Radseitenreibungsverluste, Wirbel- und Wandreibungsverluste, Spaltverluste an den Dichtflächen, Ventilverluste sowie Lagerreibungsverluste.

7.2.6. Anlagenkennlinie

Alle Pumpenanlagen besitzen eine Anlagenkennlinie $H_A = f(\dot{V}, d)$, die sich aus der Bernoulli-Gleichung ergibt:

$$H_A = \frac{p_D - p_S}{g \cdot \rho} + \frac{c_D^2 - c_S^2}{2g} + H_{\text{geo}} + H_V \tag{7.9}$$

Bei der Verwendung eines offenen Behälters entfällt der erste Term, da der Umgebungsdruck konstant ist. Die Gleichung vereinfacht sich dann zu:

$$H_A = \frac{c_D^2 - c_S^2}{2g} + H_{\text{geo}} + H_V \tag{7.10}$$

[Sur21]

7.3. Comet Tauchpumpe ELEGANT 12V

Die COMET ELEGANT 12V ist eine kompakte, leistungsfähige Niedervolt-Tauchpumpe für sauberes Wasser, auch geeignet für Trinkwasser. Sie ist dauerlaufge-eignet und ist für den Betrieb mit einem 12V DC-Netzteil oder Akku ausgelegt. [Com04] [Con25]

7.4. Spezifikation

7.4.1. Technische Daten:

- Betriebsspannung: 12 Volt DC
- Stromaufnahme: max. 2,2 A
- Leistungsaufnahme: 15 – 24 W
- Förderleistung: max. 10 l/min
- Förderhöhe: max. 5,5 m
- Fördermenge: max. 600 l/h
- Förderdruck: max. 0,55 bar
- Durchmesser: 38 mm
- Höhe: 104 mm
- Kabellänge: 1 m
- Anschlussfarben: Braun (+) / Blau (–)

7.4.2. Weitere Eigenschaften

- Trockenlaufsicher bis zu 2 Stunden
- Dauerlaufgeeigent mit einer Lebensdauer von ca. 450-500 Betriebsstunden
- Wasserverträglichkeit bis zu 60°, auch für Trinkwasser geeignet
- Passend für Schläuche mit 10 mm Innendurchmesser
- Optional erweiterbar mit: Rückschlagventil mit Entlüftung, Entlüftungsstutzen, Feinfilter

[Com04] [Con25]

7.5. Einfacher Code

Ein einfacher Beispielcode ist eine Wasserstandkontrolle. Es wird eine Meldung ausgegeben, wenn der Wasserstand zu niedrig ist. 7.1

Listing 7.1.: Einfacher Code zur Ansteuerung einer Pumpe

```
/**
 * @file ansteuerungPumpe.ino
 * @brief Control of the COMET submersible pump ELEGANT 12V using a
    MOSFET and an Arduino Nano 33 BLE Sense.
 *
 * The pump is powered by an external 12V power supply and switched via
    an N-channel MOSFET.
 * The control signals (start/stop) are given via a digital Arduino
    output pin.
 */

/// GPIO pin on the Arduino to control the MOSFET gate
const int PUMP_PIN = 2;

/// Optional maximum pump runtime in milliseconds (e.g., 60 seconds)
const unsigned long MAX_RUNTIME_MS = 60000;

/// Internal variable for time tracking
unsigned long pumpStartTime = 0;

/**
 * @brief Initializes the pump control pin as output and ensures the
    pump is off at startup.
 */
void setup() {
  pinMode(PUMP_PIN, OUTPUT);
  stopPump();  // Ensure the pump is off at startup
}

/**
 * @brief Main program loop with a simple start/stop test.
 */
void loop() {
  // Example: Start the pump for a defined time, then stop
  startPump();
  delay(10000); // Run for 10 seconds
  stopPump();
  delay(10000); // Pause for 10 seconds
}

/**
 * @brief Activates the pump by setting the MOSFET gate to HIGH.
 * Also stores the start time for monitoring.
 */
void startPump() {
  digitalWrite(PUMP_PIN, HIGH);
  pumpStartTime = millis();
}

/**
 * @brief Deactivates the pump by setting the MOSFET gate to LOW.
 */
void stopPump() {
  digitalWrite(PUMP_PIN, LOW);
  pumpStartTime = 0;
}

/**
 * @brief Returns whether the pump is currently active.
 * @return true if the pump is running, false otherwise.
 */
bool isPumpActive() {
  return digitalRead(PUMP_PIN) == HIGH;
}
```

../../Code/Arduino/Pumpe/ansteuerungPumpe/ansteuerungPumpe.ino

7.6. Tests

7.7. Einfache Anwendung

Eine Anwendung ist die Fehlererkennung für den Fall, dass kein Wasser durch die Pumpe befördert wird. Ein Ultraschallsensor misst vor dem Start der Pumpe den Wasserstand. Nach dem Start wird der Wasserstand erneut gemessen. Ändert sich der Wasserstand nicht, gibt es eine Meldung.7.2

Listing 7.2.: Einfacher Code zur Fehlererkennung

```
/**
 * @file fehlererkennungPumpe.ino
 * @brief Control of the COMET pump with water level monitoring and delivery verification.
 *
 * Detects whether water is actually being pumped while the pump is running
 * (e.g., due to air intake, blockage, or dry running).
 */

const int PUMP_PIN = 2;
const int TRIG_PIN = 3;
const int ECHO_PIN = 4;

const int MIN_WATER_LEVEL_CM = 15;        ///< Minimum water level required to start the pump
const int DELIVERY_TOLERANCE_CM = 1;      ///< Minimum level change expected during pumping
const unsigned long MEASUREMENT_INTERVAL = 3000;
const unsigned long DELIVERY_TEST_TIME = 3000; ///< Max time allowed for pump test

unsigned long lastMeasurementTime = 0;
bool pumpIsRunning = false;

/**
 * @brief Initializes the GPIO pins and serial communication.
 */
void setup() {
  pinMode(PUMP_PIN, OUTPUT);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  stopPump();

  Serial.begin(9600);
  while (!Serial);
  Serial.println("System start with error monitoring.");
}

/**
 * @brief Main loop that checks water level and verifies pump delivery.
 */
void loop() {
  if (millis() - lastMeasurementTime >= MEASUREMENT_INTERVAL) {
    lastMeasurementTime = millis();

    int levelBefore = measureWaterLevel();
    Serial.print("Water level (before): ");
    Serial.print(levelBefore);
    Serial.println(" cm");

    if (levelBefore > MIN_WATER_LEVEL_CM) {
      Serial.println("WARNING: Water level too low.");
      stopPump();
      return;
    }

    // Start pump and allow test run
    startPump();
    delay(DELIVERY_TEST_TIME);

    int levelAfter = measureWaterLevel();
    Serial.print("Water level (after): ");
    Serial.print(levelAfter);
    Serial.println(" cm");

    // Check for significant change in water level
    if (abs(levelAfter - levelBefore) < DELIVERY_TOLERANCE_CM) {
      Serial.println("ERROR: No water delivered! Please check the system.");
      stopPump();
    } else {
      Serial.println("Pump is operating correctly.");
    }

    delay(5000); // Pause before next cycle
  }
}
```

7.8. Further Readings

8. Schaltnetzteil: MW RD-50A

8.1. Einleitung

Netzteile sind elektronische Bauteile welche dazu genutzt werden, eine Eingangsspannung (oder auch Strom) in eine gewünschte Ausgangsspannung (oder Strom) umzuwandeln. Sie sind besonders wichtig für eine stabile und sichere Versorgungsspannung elektrischer Bauteile. Übergreifend lassen sich Netzteile zu den Spannungswandlern zuordnen. [Her+24]

8.2. Grundlegende Informationen

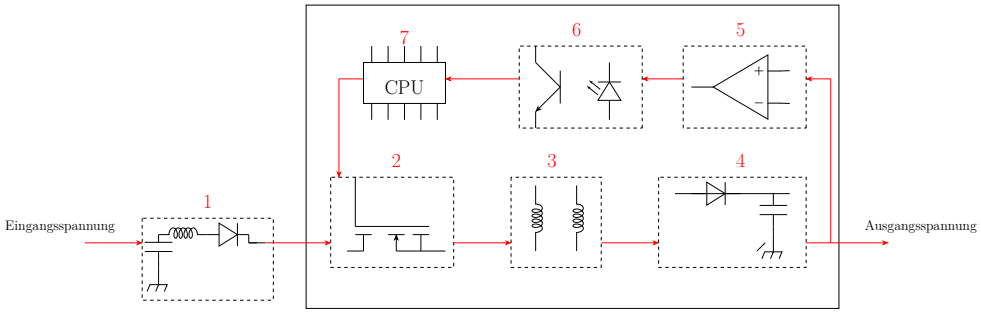
Ein Netzteil wandelt in den meisten Fällen eine bekannte Wechselspannung, wie die Netzspannung im deutschen Netz (230 V / 50 Hz), in eine gewünschte Gleichspannung um. Wechselspannungen lassen sich leichter als Gleichspannungen über weite Strecken transportieren, jedoch wird für die meisten elektrischen Geräte eine Gleichspannung benötigt. Man teilt Netzteile in 2 Gruppen ein: [Her+24]

1. **Trafonetzteile:** groß und schwer, bestehen aus Transformator, Gleichrichter und Glättungskondensator

2. **Schaltnetzteile:** klein und leicht, sehr effizient, nutzen Halbleitertechnik

Für die meisten elektronischen Bauteile werden heutzutage nur noch Schaltnetzteile eingesetzt. Sie sind leichter, effizienter, günstiger und deutlich kleiner. Im Folgenden wird daher vertieft auf das Schaltnetzteil eingegangen.[HI18] Der Aufbau eines Schaltnetzteils ist wie folgt: (vgl. 8.1) Die Eingangs-Netzfrequenz von 230 V wird im Netzgleichrichter gleichgerichtet und gesiebt. Der Schalter ist ein Halbleiterbauelement (Hochfrequenzschalter wie MOSFET) welcher aus der Gleichspannung eine Rechteckwechselspannung mit hoher Schaltfrequenz umwandelt. Der Transformator ist ein klassischer Ferritkern-Trafo und dient zur Spannungsübersetzung und zur galvanischen Trennung der Stromkreise. Im Ausgangsgleichrichter wird aus der hochfrequenten Wechselspannung eine eine kleine Gleichspannung gleichgerichtet und gesiebt. Der Regelverstärker dient zur Verstärkung der Ausgangsspannung und führt diese in den eingebauten Regelkreis des Schaltnetzteils. Um den Regelkreis und die empfindliche Elektronik vom Ausgangsstromkreis zu trennen, wird eine Potentialtrennung (Optokoppler) verwendet. Die Steuer- und Überwachungsschaltung wird mit einem Microcontroller (CPU) realisiert. Dieser überwacht und steuert den Regelkreis und sorgt somit für eine konstante und stabile Ausgangsspannung. [HI18]

Figure 8.1.: Aufbau eines Schaltnetzteils



8.3. Schaltnetzteil: MW RD-50A

Das Schaltnetzteil in unserem Projekt ist das „MeanWell RD-50A“. Dabei handelt es sich um ein Schaltnetzteil mit variabler Eingangsspannung und zwei Ausgangsspannungen: 5 V und 12 V. Als Eingangsspannung wird die Netzspannung von 230 V und 50 Hz genutzt, das Netzteil stellt daraufhin die Versorgungsspannung für Arduino und Sensor (5 V) her und ebenfalls die Lastspannung von 12 V für die Pumpe. [Mea] [Rei25d]

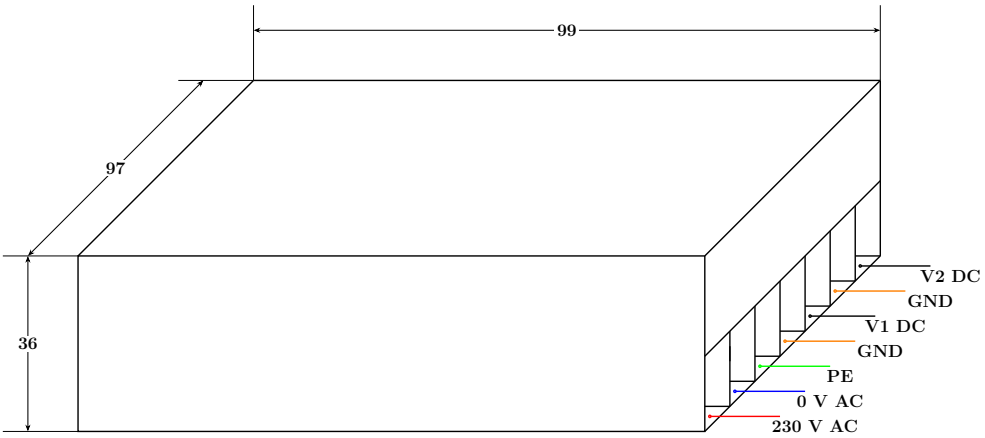
8.4. Spezifikation

Das MW RD-50A, zu sehen in 8.2, verfügt über eine variable Ausgangsspannung von 5/12 V und eine Ausgangsleistung von 54 W bei einem Ausgangsstrom von 6 A. Das Schaltnetzteil hat einen Wirkungsgrad von 79% und verfügt über einen integrierten Kurzschlussschutz, Überlastschutz und einen Überspannungsschutz. Es hat mit seiner kompakten Bauform ein Gewicht von 410 g und ist verfügt über Schraubanschlüsse. In der Tabelle 8.1 ist die Pinbelegung zu sehen. Die Pins 1-3 sind die L1 Phase vom AC Netz, Neutralleiter und PE. Sie bilden die Versorgungsspannung. Die Pins 4 und 6 sind für die Erdung der beiden Gleichstromkreise zuständig. Pin 5 sorgt für die 5 V DC Spannung und Pin 7 ist für das 12 V DC Netz zuständig.[Mea] [Rei25d]

Table 8.1.: Pinbelegung des Schaltnetzteils

1	L1 – 230 V AC
2	N – 0 V AC
3	PE
4	GND (DC)
5	V1 – 5 V DC
6	GND DC
7	V2 – 12 V (DC)

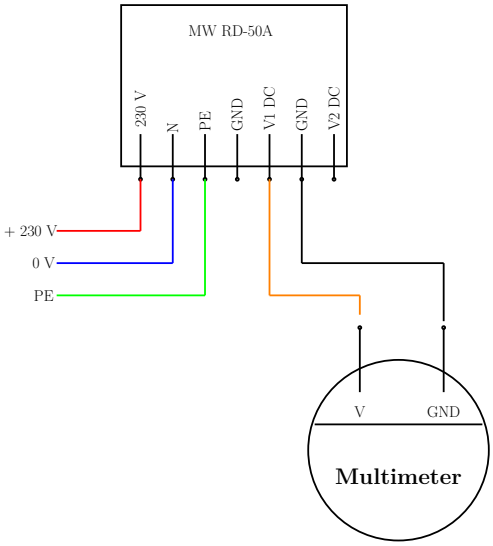
Figure 8.2.: Skizze des Netzteils



8.5. Tests

Mit einem einfachen Multimeter kann man die Ausgangsspannungen des Schaltnetzteils überprüfen. Dazu schließt man das Netzteil am AC-Netz an und verbindet die Messspitzen des Multimeters mit dem GND und jeweils dem V1 und V2 Anschluss, sieh dazu 8.3. Am Multimeter muss jedoch das Wahlrad auf DC 20 V eingestellt werden. Zu beachten ist, dass das Netzteil eine Toleranz der Ausgangsspannung von ± 2% hat und das Multimeter auch eine interne Messabweichung hat. Dies ist zu berücksichtigen, daher empfiehlt es sich, eine Versuchsreihe von mindestens 3 Versuchen durchzuführen und das arithmetische Mittel der 3 Anzeigewerte zu nehmen.

Figure 8.3.: Spannungsmessung des Netzteils

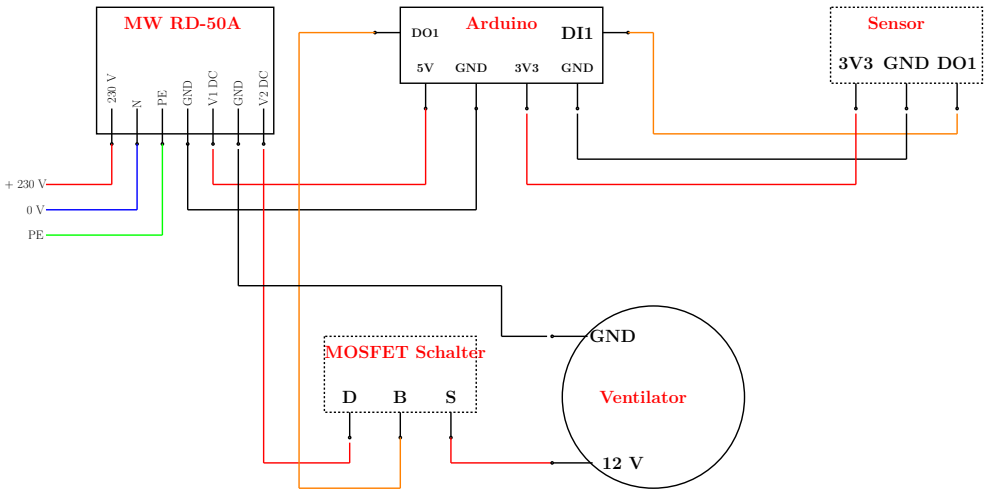


8.6. Einfache Anwendung

Als eine einfache Anwendung kann man zum Beispiel das Netzteil verwenden, um einen Steuer- und einen Laststromkreis aufzubauen. Mit dem Steuerstromkreis werden Sensoren und Mikrocontroller versorgt und der Laststromkreis ist für größere Aktoren wie ein Ventilator zuständig. In diesem Anwendungsfall (8.4) ist der Mikrocontroller

ein Arduino und der Sensor ein Luftfeuchtigkeitssensor. Sinkt die Luftfeuchtigkeit im Raum registriert dies der Sensor. Der Arduino schaltet dann den Laststromkreis mit einem MOSFET frei und der Ventilator beginnt sich zu drehen. Mit folgendem Code lässt sich das realisieren: 8.1

Figure 8.4.: Anwendung des Netzteils für 2 seperate Stromkreise



Listing 8.1.: Einfacher Code zum Verwenden des Netzteils

```
/**
 * @file anwendungSchaltnetzteil.ino
 * @brief Controls a 12V fan via a MOSFET based on room humidity.
 *
 * If the room humidity drops below 40%, the Arduino activates DO1,
 * which switches a MOSFET to power a 12V fan.
 *
 * @date 2025-04-28
 */

#include <DHT.h>

/// Digital pin connected to the DHT sensor
#define DHT_PIN 2

/// Digital pin to control the MOSFET (DO1)
#define FAN_CONTROL_PIN 3

/// DHT sensor type: DHT11 or DHT22
#define DHT_TYPE DHT22

/// Humidity threshold in percent
#define HUMIDITY_THRESHOLD 40.0

/// DHT sensor instance
DHT dht(DHT_PIN, DHT_TYPE);

/**
 * @brief Arduino setup function.
 * Initializes serial communication, DHT sensor, and fan control pin.
 */
void setup() {
  Serial.begin(9600);
  dht.begin();
  pinMode(FAN_CONTROL_PIN, OUTPUT);
  digitalWrite(FAN_CONTROL_PIN, LOW); // Ensure fan is off at startup
}

/**
 * @brief Arduino main loop.
 * Reads humidity and activates fan if below threshold.
 */
void loop() {
  float humidity = dht.readHumidity();

  // Check if sensor reading is valid
  if (isnan(humidity)) {
    Serial.println("Failed to read humidity from sensor.");
    return;
  }

  Serial.print("Humidity: ");
  Serial.print(humidity);
  Serial.println(" %");

  // Control fan based on humidity
  if (humidity < HUMIDITY_THRESHOLD) {
    digitalWrite(FAN_CONTROL_PIN, HIGH);  ///< Turn on fan
    Serial.println("Fan ON (humidity below threshold)");
  } else {
    digitalWrite(FAN_CONTROL_PIN, LOW);   ///< Turn off fan
    Serial.println("Fan OFF (humidity above threshold)");
  }

  delay(2000); // Wait before next reading
}
```

../../Code/Arduino/Schaltnetzteil/anwendungSchaltnetzteil/anwendungSchaltnetzteil.ino

8.7. Further Readings

Tränkler, H.-R.:Sensortechnik – Handbuch für Praxis und Wissenschaft. 2. Aufl., Springer Vieweg, 2018 [TR18]

- Ein schöner Einstieg und Überblick über die Sensortechnik und ihre Anwendung heutzutage.

E. Hering et al. *Elektrotechnik und Elektronik in Maschinenbau und Mechatronik*. 5. Auflage. Springer, 2024. ISBN: 978-3-662-67538-0.[Her+24]

- Eine sehr spannende Einführung in die Elektrotechnik für Maschinenbauer. Viele grundlegende Informationen und gute Anwenderbeispiele.

Haeberle and Isele. Tabellenbuch Elektrotechnik. 28th ed. Europa Lehrmit- tel, 2018. isbn: 978-3-808-53490-8.[HI18]

- Ein tolles Tabellenbuch in dem viel Grundlagenwissen vermittelt wird. Zudem sind die einfache Skizzen und Erklärungen hervorzuheben.

9. Bluetooth Modul HC-05

9.1. Einleitung

Bluetooth-Module sind elektronische Komponenten, die drahtlose Kommunikation zwischen Geräten ermöglichen. Sie wandeln elektrische Signale in Funkwellen um und sorgen so für eine zuverlässige Datenübertragung über kurze Distanzen. Grundsätzlich unterscheidet man Bluetooth Classic (für kontinuierliche Verbindungen wie Audio-Streaming) und Bluetooth Low Energy (BLE) (energiesparend für Geräte mit begrenzter Akkulaufzeit). Ihre Hauptaufgabe ist es, Geräte wie Kopfhörer, Sensoren oder Smart-Home-Systeme einfach und kabellos zu vernetzen – ohne komplexe Infrastruktur [Sau22].

9.2. Grundlegende Informationen

Bluetooth-Module sind spezialisierte Hardwarekomponenten, die eine drahtlose Kommunikation zwischen Geräten über kurze Distanzen ermöglichen. Typischerweise decken sie Reichweiten von 10 bis 30 Metern ab, abhängig von Umgebungsfaktoren wie Wänden oder Störquellen. Im Kern bestehen diese Module aus einem Bluetooth-Chip, einer Antenne sowie unterstützender Elektronik wie Spannungsreglern oder Schnittstellen-ICs. Um Platz und Energie zu sparen, werden sie häufig als System-on-a-Chip (SoC) realisiert, bei dem alle notwendigen Komponenten in einem einzigen, kompakten Bauteil integriert sind. Die Technologie nutzt das 2,4-GHz-ISM-Band (2400–2483,5 MHz), ein lizenzfreies Frequenzspektrum, das auch von WLAN, Mikrowellen und anderen Funkstandards genutzt wird. Um Interferenzen zu vermeiden, setzt Bluetooth auf Frequency Hopping: Das Modul wechselt bis zu 1600-mal pro Sekunde zwischen 79 definierten Kanälen, wobei jeder Kanal eine Bandbreite von 1 MHz besitzt. Die genaue Frequenz eines Kanals k.
k lässt sich durch die Gleichung

$$f_k = 2402 + k\ [MHz] \tag{9.1}$$

berechnen, wobei k Werte von 0 bis 78 annimmt. Dieser dynamische Kanalwechsel sorgt für eine robuste Verbindung, selbst in frequenzbelasteten Umgebungen [Sau22].

9.2.1. Bluetooth Varianten

Grundsätzlich unterscheidet man zwei Bluetooth-Varianten: Bluetooth Classic und Bluetooth Low Energy (BLE).

- Bluetooth Classic ist für Anwendungen mit hohen Datenraten ausgelegt, etwa Audio-Streaming für Kopfhörer oder Lautsprecher, und erreicht bis zu 3 Mbps (Megabit pro Sekunde) durch Erweiterungen wie EDR (Enhanced Data Rate).

- BLE hingegen wurde für Geräte entwickelt, die extrem energieeffizient arbeiten müssen, wie Wearables oder Sensoren in IoT-Netzwerken. Hier liegt die maximale Datenrate bei 1 Mbps, während der Stromverbrauch im Standby-Modus oft unter 15 µA bleibt – ideal für Batteriebetrieb über Monate oder Jahre.

9.2.2. Reichweite und Sendeleistung

Die Reichweite eines Bluetooth-Moduls hängt maßgeblich von seiner Sendeleistung (TX Power) ab.

- Klasse 1 Geräte (bis 100 mW Sendeleistung) können theoretisch bis zu 100 m überbrücken.
- Klasse 2 Geräte (2,5 mW Sendeleistung) sind auf etwa 10 m begrenzt.
- Klasse 3 Geräte (1 mW Sendeleistung) sind auf etwa 1 m begrenzt.

In der Praxis reduzieren Hindernisse wie Wände oder elektronische Störquellen diese Werte jedoch deutlich [Sau22].

9.2.3. Daten Übertragung

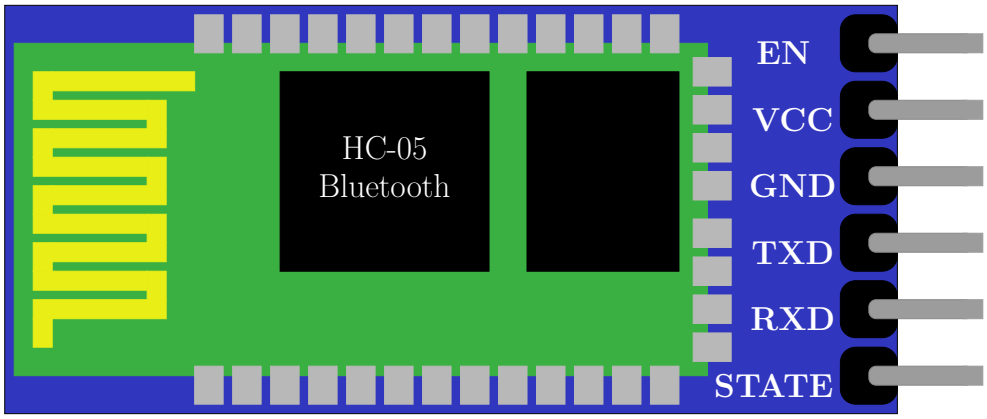
Ein kritischer Aspekt ist die Sicherheit. Bluetooth verwendet AES-128-Verschlüsselung, um Daten vor unbefugtem Zugriff zu schützen. Beim Verbindungsaufbau (Pairing) kommen zudem Protokolle wie Elliptic Curve Diffie-Hellman (ECDH) zum Einsatz, die einen sicheren Schlüsselaustausch ermöglichen und Man-in-the-Middle-Angriffe erschweren. Dank standardisierter Profile – vordefinierter Regeln für spezifische Anwendungen – lassen sich Geräte unterschiedlicher Hersteller problemlos koppeln. Beispiele sind das *A2DP-Profil* für Audio-Streaming oder das HID-Profil für Tastaturen und Mäuse [Sau22].

9.2.4. Anwendungen

Bluetooth-Module sind heute in nahezu allen Consumer-Geräten zu finden, von Smartphones über Smart-Home-Steuerungen bis hin zu industriellen Sensornetzwerken. Ihre Stärke liegt in der einfachen Integration: Über Schnittstellen wie UART oder SPI können sie direkt mit Mikrocontrollern wie Arduino oder Raspberry Pi verbunden werden, um Projekte schnell drahtlos zu machen. Dennoch stößt die Technologie an Grenzen, wenn es um hohe Reichweiten, extrem hohe Datenmengen (z. B. Video-Streaming) oder komplexe Mesh-Netzwerke geht – hier sind Alternativen wie WLAN oder Zigbee oft besser geeignet. Zusammenfassend bilden Bluetooth-Module das technische Rückgrat für unzählige kabellose Anwendungen des täglichen Lebens. Sie kombinieren Energieeffizienz mit standardisierter Kompatibilität und bleiben dabei für Nutzer weitgehend unsichtbar. Mathematische Grundlagen wie Frequenzberechnung, Verschlüsselungsalgorithmen oder Leistungsbilanzierung sorgen im Hintergrund für Zuverlässigkeit, während Anwender lediglich die simplicity der „Plug-and-Play“-Vernetzung schätzen [Sau22].

9.3. Bluetooth Modul: HC-05

Figure 9.1.: Skizze des Bluetooth Moduls HC-05



Das HC-05 Bluetooth-Modul, zu sehen in 9.1, ist ein serielles Kommunikationsmodul, das auf der Bluetooth-Spezifikation V2.0+EDR basiert und für drahtlose Datenübertragung über das Serial Port Protocol (SPP) konzipiert ist. Es ermöglicht sowohl Master- als auch Slave-Betriebsmodi und wird häufig in Embedded-Systemen zur Integration von Bluetooth-Funktionalität eingesetzt [**HC05Datasheet:2010**]

9.4. Spezifikation

??????????????????????????????ß

9.4.1. Hardware-Spezifikation

Die Kommunikationsfähigkeiten des HC-05-Moduls bilden die Grundlage seiner Einsatzflexibilität in drahtlosen Systemen. Als Bluetooth-V2.0+EDR-Modul nutzt es das 2,4-GHz-ISM-Band, das weltweit für industrielle, wissenschaftliche und medizinische Anwendungen lizenzfrei genutzt werden kann. Die Kombination aus GFSK-Modulation (Gaussian Frequency Shift Keying) und Adaptive Frequency Hopping (AFH) gewährleistet eine robuste Datenübertragung selbst in frequenzbelasteten Umgebungen. AFH reduziert Interferenzen durch dynamische Kanalwechsel, was besonders in Umgebungen mit mehreren Funkgeräten kritisch ist.
Im Detail umfassen die Leistungsmerkmale:

1. Protokoll und Datenraten
   - Bluetooth V2.0+EDR (Enhanced Data Rate) mit einer maximalen Datenrate von 3 Mbps im synchronen Modus und 2,1 Mbps im asynchronen Modus [**ReicheltHC05:2025**][**HC05Datasheet:2010**].
   - Unterstützung des Serial Port Profiles (SPP) für bidirektionale Master-/Slave-Kommunikation [**ReicheltHC05:2025**].
2. Sende- und Empfängerleistung
   - Sendeleistung: +4 dBm (Klasse 2), geeignet für Reichweiten bis zu 10 m in offenen Räumen (Quelle 2, S. 1).
   - Empfindlichkeit: -80 dBm (typisch) bzw. -84 dBm bei einer Bitfehlerrate (BER) von 0,1 % – ein Wert, der eine zuverlässige Verbindung auch bei schwachen Signalen ermöglicht [**HC05Datasheet:2010**][**ReicheltHC05:2025**].

Diese Parameter positionieren das HC-05-Modul als geeignete Lösung für Anwendungen wie Industrieautomation, IoT-Sensornetzwerke und Consumer-Geräte, bei denen Zuverlässigkeit und moderate Reichweite im Vordergrund stehen.

9.4.2. Elektrische Eigenschaften

Die elektrische Integration des Moduls erfordert eine präzise Abstimmung der Spannungsversorgung und Stromaufnahme, um Kompatibilität mit gängigen Mikrocontrollern zu gewährleisten. Das Modul ist speziell für Low-Power-Anwendungen optimiert, was durch folgende Merkmale erreicht wird:

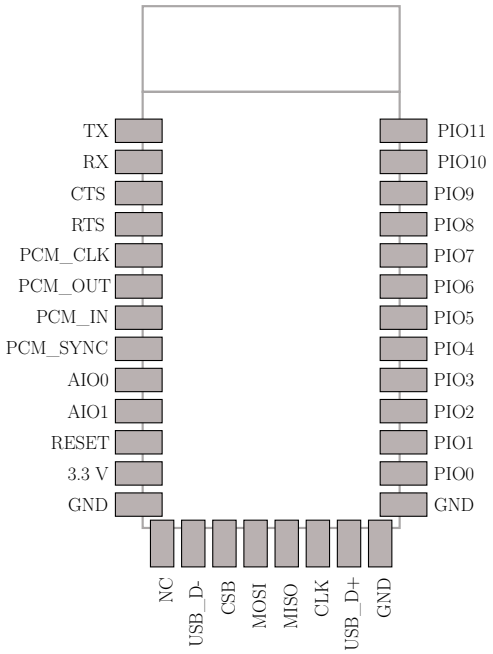
1. Spannungsversorgung
   - Betriebsspannung: +3,3 V DC (±5 %), versorgt durch einen integrierten Linearregler, der Eingangsspannungen bis 5 V toleriert [**ReicheltHC05:2025**].
   - I/O-Spannungspegel: 1,8–3,6 V, ermöglicht die direkte Anbindung an 3,3-V- und 5-V-Systeme ohne zusätzliche Pegelwandler [**HC05Datasheet:2010**].
2. Stromverbrauch
   - Typischer Betriebsstrom: 50 mA im aktiven Modus [**ReicheltHC05:2025**].
   - Energiesparmodi: Programmierbare SNIFF-Modi reduzieren den Verbrauch durch intermittierende Aktivität, ideal für batteriebetriebene Geräte [**HC05Datasheet:2010**].

Diese Eigenschaften machen das HC-05-Modul zu einer energieeffizienten Option für Projekte mit begrenzten Stromressourcen, wie tragbare Geräte oder Fernsensoren.

9.4.3. Schnittstellen und Anschlüsse

Die Flexibilität des HC-05-Moduls zeigt sich in seiner Unterstützung für multiple Schnittstellen, die eine nahtlose Integration in bestehende Systemarchitekturen ermöglichen.

Figure 9.2.: Schnittstellen und Anschlüsse HC-05



1. UART-Schnittstelle:
   - Standardkonfiguration: 38400 Baud, 8 Datenbits, 1 Stoppbit, keine Parität – eine Einstellung, die mit den meisten Mikrocontrollern kompatibel ist [**HC05Datasheet:2010**].
   - Programmierbare Baudraten: Anpassbar von 2400 bis 1382400 Baud, um höhere Datenraten oder spezielle Protokolle zu unterstützen [**ReicheltHC05:2025**].
   - Pin-Details
     - UART_RX (Pin 2): Empfängt serielle Daten (CMOS-Eingang mit Pull-Down-Widerstand zur Vermeidung von Floating-Signalen) [**HC05Datasheet:2010**].
     - UART_TX (Pin 1): Sendet serielle Daten (CMOS-Ausgang mit Pull-Up-Widerstand für stabile Logikpegel) [**HC05Datasheet:2010**].
2. SPI-Schnittstelle
   - Ermöglicht die Anbindung an Peripheriegeräte oder Mikrocontroller mit SPI-Bus.
   - Pins:
     - SPI_CLK (Pin 19): Taktsignal für die SPI-Kommunikation [**HC05Datasheet:2010**].
     - SPI_MISO (Pin 18) und SPI_MOSI (Pin 17): Datenleitungen für Eingabe und Ausgabe [**HC05Datasheet:2010**].
     - SPI_CSB (Pin 16): Chip-Select-Pin (aktiv niedrig) zur Steuerung der Geräteauswahl [**HC05Datasheet:2010**].
3. GPIO-Pins:
   - 12 programmierbare I/O-Pins (PIO0–PIO11):
     - PIO10 (Pin 33) und PIO11 (Pin 34): Steuern die Status-LEDs (rot/blau), deren Blinkmuster Verbindungszustände wie Paarung oder Trennung anzeigen [**HC05Datasheet:2010**].
     - PIO0 (Pin 23) und PIO1 (Pin 24): Optional zur Ansteuerung externer RF-Komponenten wie LNA (Low-Noise Amplifier) oder PA (Power Amplifier) [**HC05Datasheet:2010**].

Diese Schnittstellenvielfalt ermöglicht Anwendungen von der einfachen seriellen Datenübertragung bis zur komplexen Steuerung externer Hardwarekomponenten.

9.4.4. Mechanische und Umgebungsbedingungen

Die mechanische Robustheit und Umgebungstoleranz des HC-05-Moduls sind entscheidend für den Einsatz in anspruchsvollen Umgebungen.

1. Abmessung:
   - Größe: 12,7 mm × 27 mm – ein kompaktes Design, das die Integration in platzbeschränkte Gehäuse erleichtert [**HC05Datasheet:2010**].
2. Umgebungsbedingung
   - Betriebstemperatur: -5 °C bis +45 °C, was den Einsatz in den meisten Indoor-Umgebungen abdeckt [**ReicheltHC05:2025**].
   - Lagerungstemperatur: Obwohl nicht explizit spezifiziert, ist das Modul durch seine Bauweise für industrielle Anwendungen mit moderaten Temperaturschwankungen geeignet [**HC05Datasheet:2010**].

Diese Spezifikationen unterstreichen die Eignung des Moduls für Anwendungen in der Automatisierungstechnik, Smart-Home-Systemen und mobilen Geräten.

9.4.5. Zusätzliche Hardware-Features

Neben den Kernfunktionen bietet das HC-05-Modul weitere Hardwaremerkmale, die seine Vielseitigkeit erhöhen:

1. Integrierte Antenne:
   - Eliminiert die Notwendigkeit externer Antennen und reduziert den Platzbedarf auf der Leiterplatte [**HC05Datasheet:2010**].
2. Reset-Funktion
   - RESETB (Pin 11): Ein aktiver Low-Impuls (>5 ms) setzt das Modul zurück, was bei Softwarefehlern oder Neukonfigurationen nützlich ist [**HC05Datasheet:2010**].
3. USB-Anschluss:
   - USB (Pin 15): Optionaler Anschluss für USB v1.1/2.0, allerdings ohne detaillierte Dokumentation in den Quellen – möglicherweise für Firmware-Updates oder alternative Kommunikationswege [**HC05Datasheet:2010**].

Diese Features tragen zur Reduzierung externer Komponenten bei und vereinfachen die Prototypenentwicklung.

9.4.6. Pinbelegung (Auszug)

Die Pinbelegung des HC-05-Moduls ist entscheidend für die korrekte Verdrahtung in Schaltungen. Die folgende Tabelle fasst die wichtigsten Pins zusammen:

Table 9.1.: Die Wichtigsten Pins des HC-05

Pin-Nr.	**Name**	**Funktion**	**Anmerkung**
1	UART_TX	Serieller Datenausgang	CMOS-Ausgang mit Pull-Up
2	UART_RX	Serieller Dateneingeng	CMOS-Eingang mit Pull-Down
12	3.3 VCC	Spannungsversorgung (+3,3 V)	Integrierter Linearregler
13,21,22	GND	Masseanschluss	Mehrere Pins zur Erdung

9.5. Bibliotheken

Um die Kommunikation zwischen Arduino und HC-05 zu ermöglichen wird die Bibliothek SoftwareSerial.h benötigt. Durch die Verwendung der Bibliothek kann der HC-05 über beliebige Pins Kommunizieren. Die Bibliothek wird hier mit der Installation der aktuellsten Version von Arduino-IDE (Version 2.2.1) mit Installiert.Falls die Bibliothek dennoch fehlen sollte kann sie wie folgt Installiert werden:

- Bibliothek über Arduino Libery List Herunterladen
- Öffen Sie die Arduino-IDE
- Gehen Sie zu Sketch > Bibliothek einbinden > .ZIP-Bibliothek hinzufügen.
- Wählen Sie die heruntergeladene ZIP-Datei aus
- Die Bibliothek wird automatisch im Ordner Documents/Arduino/libraries/ installiert

Optional kann man noch die Bibliothek HC05Libary Installieren mit dieser Bibliothek wird das Senden von AT-Befehlen vereinfacht.

Installation: Download:

- GitHub: https://github.com/evankale/ArduinoHC05

- Klicke auf Code > Download ZIP.

- In der Arduino-IDE: Sketch > Bibliothek einbinden > .ZIP-Bibliothek hinzufügen.

9.6. Einfacher Code

Um die grundlegende Funktionsweise des HC-05 Bluetooth-Moduls zu testen kann der folgende Code verwendet werden. Der Code überprüft ob das HC-05 mit dem Smartphone verbunden ist und ob die Übertragung von Daten Funktioniert. Um das Smartphone mit dem HC-05 zu verbinden wird ebenfalls ein Code benötigt dieser ist falls er nicht verändert wurde: 1234 oder 0000 falls der Code geändert wurde kann dieser über ein I2C und einem entsprechenden Code ausgelesen werden. 9.1

Listing 9.1.: Code zur Überprüfung der Funktionalität

```
/**
 * @file bluetoothBeispiel.ino
 * @brief Basic Bluetooth connection indicator using internal LED.
 *
 * This sketch lights up the internal LED (pin 13) when a Bluetooth
    connection is active.
 * It also sends a confirmation message "OK!" over Bluetooth.
 */

#include <SoftwareSerial.h> ///< Include SoftwareSerial library for
    Bluetooth communication

/// Create a SoftwareSerial object for Bluetooth communication (RX = pin
     2, TX = pin 3)
SoftwareSerial bluetooth(2, 3);

/**
 * @brief Arduino setup function.
 *
 * Sets pin 13 as output and initializes Bluetooth communication.
 */
void setup() {
  pinMode(13, OUTPUT);        ///< Use internal LED (pin 13) as output
  bluetooth.begin(9600);      ///< Start Bluetooth communication at 9600
    baud
}

/**
 * @brief Arduino main loop function.
 *
 * Turns the internal LED on when data is available from the Bluetooth
    module,
 * sends a response message, and prevents message spamming with a short
    delay.
 */
void loop() {
  if (bluetooth.available()) {      ///< Check if Bluetooth data is
    available
    digitalWrite(13, HIGH);         ///< Turn on internal LED
    bluetooth.println("OK!");       ///< Send response message
    delay(1000);                    ///< Wait to avoid message flooding
  } else {
    digitalWrite(13, LOW);          ///< Turn off internal LED if no
    Bluetooth data
  }
}
```

../../Code/Arduino/Bluetooth Modul/bluetoothBeispiel/bluetoothBeispiel.ino

9.7. Tests

9.8. Einfache Anwendung

Als Beispielhafte Anwendung für den HC-05 kann das Modul über einen Arduino und einem Relais mit einer Stehlampe verschaltet werden. Zu sehen in folgendem Code: 9.2. Wenn man das System dann mit einem Smartphone verbindet kann die Stehlampe über das System mit den HC-05 ein und ausgeschaltet werden. Das System kann folgenderweise verbunden werden 9.2:

Die Stehlampe wird dann noch mit dem Relais verbunden. Dann kann auch schon der Code für die Anwendung Installiert werden.

Table 9.2.: Pin Belegung

Arduino	**HC-05**	**Realis**
RX(Pin 2)	TX	-
TX(Pin 3)	RX	-
5V (oder 3.3V)	VCC	-
GND	GND	-
Pin 7	-	IN(Steuereingang)

Listing 9.2.: Code zur Überprüfung der Funktionalität

```
/**
 * @file bluetoothAnwendung.ino
 * @brief Bluetooth-controlled relay for switching a floor lamp on and off.
 *
 * This sketch uses a Bluetooth module connected via SoftwareSerial to control a relay.
 * Commands 'E' and 'A' turn the relay ON and OFF, respectively.
 */

#include <SoftwareSerial.h> ///< Include SoftwareSerial library for Bluetooth communication

#define RELAY_PIN 7 ///< Arduino pin connected to the relay module

/// Create a SoftwareSerial object for Bluetooth communication (RX = pin 2, TX = pin 3)
SoftwareSerial bluetooth(2, 3);

/**
 * @brief Arduino setup function.
 *
 * Initializes the relay pin and starts the Bluetooth serial communication.
 * Sends an initial message to the Bluetooth device to indicate readiness.
 */
void setup() {
  pinMode(RELAY_PIN, OUTPUT);        ///< Set the relay pin as output
  digitalWrite(RELAY_PIN, LOW);      ///< Start with the relay turned OFF
  bluetooth.begin(9600);             ///< Start Bluetooth communication at 9600 baud
  bluetooth.println("Floor lamp ready! Send 'E' to turn ON, 'A' to turn OFF.");
}

/**
 * @brief Arduino main loop function.
 *
 * Continuously checks for incoming Bluetooth commands and toggles the relay accordingly.
 */
void loop() {
  if (bluetooth.available()) {
    char cmd = bluetooth.read(); ///< Read the incoming command character

    if (cmd == 'E') {
      digitalWrite(RELAY_PIN, HIGH); ///< Turn the relay ON
      bluetooth.println("Floor lamp ON");
    }
    else if (cmd == 'A') {
      digitalWrite(RELAY_PIN, LOW); ///< Turn the relay OFF
      bluetooth.println("Floor lamp OFF");
    }
  }
}
```

../../Code/Arduino/Bluetooth Modul/bluetoothAnwendung/bluetoothAnwendung.ino

9.9. Further Readings

10. Kippschalter: GOOBAY 10013

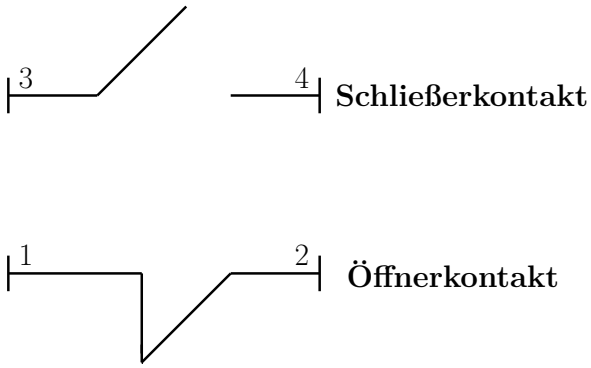
10.1. Einleitung

Kippschalter sind elektrische Bauteile, die durch ihre einfache Handhabung und Vielseitigkeit in elektronischen Systemen und Anlagen ihre Verwendung finden. Sie ermöglichen das Ein- und Ausschalten von Stromkreisen oder das Umschalten zwischen verschiedenen Betriebsmodi. [NKK25] [Ind25]

10.2. Grundlegende Informationen

Ein Kippschalter besteht typischerweise aus einem Hebel, der mechanisch zwei oder mehr elektrische Kontakte verbindet oder trennt. Beim Betätigen des Hebels bewegt sich ein beweglicher Kontakt, der entweder einen Stromkreis schließt oder öffnet. Diese mechanische Bewegung ermöglicht eine klare und stabile Schaltposition, was Kippschalter besonders zuverlässig macht. Je nach Ausführung können Kippschalter ein- oder mehrpolig sein und unterschiedliche Schaltfunktionen wie EIN/AUS, EIN/EIN oder EIN/AUS/EIN realisieren. Intern können sie entweder als Schließer oder Öffner funktionieren, zu sehen in 10.1.[NKK25] [Ind25]

Figure 10.1.: Skizze der Schaltertypen



10.3. Schalter: GOOBAY 10013

Der Goobay Miniatur-Kippschalter ist ein kompakter Schalter mit drei Pins und der Schaltfunktion EIN–AUS–EIN. Diese Konfiguration ermöglicht es, zwischen zwei Stromkreisen umzuschalten oder einen Stromkreis vollständig zu unterbrechen. Der Schalter ist ideal für Anwendungen im DIY-Bereich, Modellbau oder in elektronischen Projekten, bei denen Platzersparnis und Zuverlässigkeit gefragt sind. [NKK25] [goo23] [Rei25b]

10.4. Spezifikation

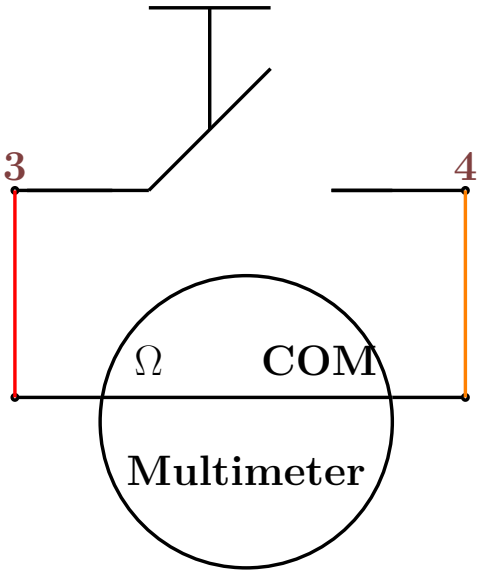
- Kippschalter mit 2 Pins (Lötösen)

- LxBxH: 8 x 5 x 7 mm
- Nennstrom: 6 A bei 125 V AC oder 3 A bei 250 V AC
- RoHS konform
- Gewicht: 3 g
- Farbe: Blau

10.5. Tests

Man kann die Funktionalität des Kippschalters leicht testen. Da er als Schließer (EIN-AUS) ausgeführt ist, kann man eine einfache Durchgangsprüfung machen, wie in 10.2 zu sehen. Man stellt das Wahlrad des Multimeters zuerst auf Widerstandsmessung. Danach schließt man das Multimeter mit den Anschlüssen für Widerstand und COM an den beiden Kontakten des Schalters an. Ist der Schalter geschlossen, zeigt das Multimeter einen hochohmigen Messwert. Schließt man jedoch den Schalter, wird der Widerstand sehr gering. Damit hat man überprüft, dass der Schalter einen Durchfluss des Stroms ermöglicht.[NKK25] [FLU25]

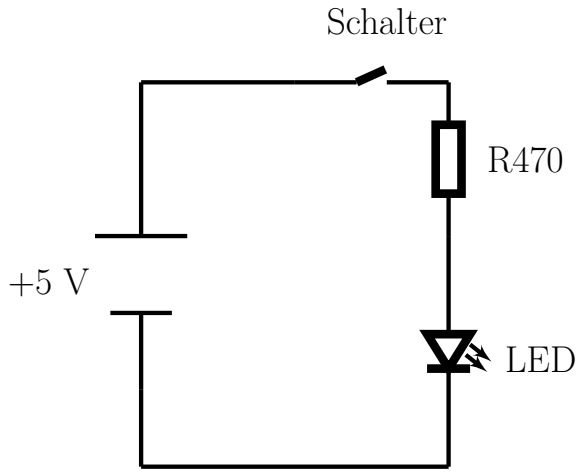
Figure 10.2.: Messen des Widerstands



10.6. Einfache Anwendung

Als einfache Anwendung kann man einen einfachen Lichtschalter betrachten. Eine Gleichstromquelle versorgt eine LED mit Vorwiderstand mit einer 5 V Spannung. Vor dem Widerstand ist ein einfacher Kippschalter als EIN-AUS eingebaut. Dieser kann den Stromkreis ein- und ausschalten. 10.3

Figure 10.3.: Einfache Anwendung des Schalters



11. MOS-FET

11.1. Einleitung

MOSFETs (Metal-Oxide-Semiconductor Field-Effect Transistors) sind spannungsgesteuerte Halbleiterbauelemente, bei denen das Gate durch eine isolierende SiO_2-Schicht vom Halbleiter getrennt ist. Sie ermöglichen das Schalten und Regeln von Strömen mit sehr geringem Steuerstrom und werden daher häufig in digitalen und analogen Schaltungen eingesetzt. [All24]

11.2. Grundlegende Informationen

Der Metall-Oxid-Halbleiter-Feldeffekttransistor (MOS-FET) ist ein spannungsgesteuerter Transistor, der in vier Grundtypen unterteilt wird: n-Kanal- und p-Kanal-Typen, jeweils als Anreicherungs- oder Verarmungstyp. Der am häufigsten verwendete Typ ist der n-Kanal-MOS-FET im Anreicherungstyp, der bei einer Gate-Source-Spannung von 0 V gesperrt ist (selbstsperrend).[All24]
Ein MOS-FET besteht aus zwei n-dotierten Bereichen – Source und Drain –, die auf einem p-dotierten Substrat liegen. Zwischen diesen Bereichen befindet sich ein isoliertes Gate. Wird eine positive Spannung am Gate angelegt, zieht sie Elektronen an und bildet einen leitfähigen Kanal zwischen Source und Drain. Die Stromleitung erfolgt dabei nahezu verlustfrei über diesen Kanal, wobei der Steuerstrom sehr gering ist, da das Gate durch eine Oxidschicht elektrisch isoliert ist.[All24]
Je nach Höhe der Gate-Source-Spannung (UGS) im Verhältnis zur Schwellenspannung (UTh) ergeben sich drei Betriebsbereiche [All24]:

- Sperrbetrieb (UGS < UTh): Kein leitender Kanal – der Transistor sperrt.
- Abschnürbereich (UGS > UTh, aber UGS - UTh < UDS): Der Kanal ist nur teilweise ausgebildet – Strom fließt eingeschränkt.
- Ohmscher Bereich (UGS - UTh > UDS): Der Kanal reicht vollständig von Source bis Drain – der Transistor verhält sich wie ein variabler Widerstand.

Die Eingangswiderstände von MOS-FETs sind extrem hoch (im $G\Omega$-Bereich), was sie ideal für Anwendungen mit geringer Steuerleistung macht. Jedoch kann bei hohen Frequenzen die Gate-Kapazität zu Nachteilen führen, da sie zu Umladeströmen führt, die wiederum die Geschwindigkeit der Schaltung begrenzen.[All24]
Die Kennlinienfelder, die das Verhalten von Drainstrom (ID) in Abhängigkeit von UGS und UDS zeigen, weisen zum Teil deutliche Unterschiede zwischen Simulation und realen Messungen auf. Diese Abweichungen entstehen u.a. durch Parameterstreuungen bei der Fertigung, insbesondere bei der Thresholdspannung und der sogenannten Kanallängenmodulation (Early-Effekt). Dies spielt in digitalen Schaltungen oft eine untergeordnete Rolle, kann jedoch in analogen Anwendungen zu Problemen führen, vor allem bei hohen Temperaturen.[All24]
Insgesamt bietet der MOS-FET viele Vorteile wie hohe Energieeffizienz und einfache Ansteuerung, erfordert aber eine sorgfältige Berücksichtigung seiner physikalischen Eigenschaften, insbesondere bei hohen Frequenzen oder großen Temperaturbereichen.[All24]

MOS-FETs lassen sich grundsätzlich in zwei Hauptgruppen unterteilen: n-Kanal- und p-Kanal-MOS-FETs. Diese Bezeichnungen beziehen sich auf die Art der Dotierung des Halbleitermaterials. Die Dotierung erfolgt entweder mit Donatoren (n-Typ, negativ) oder Akzeptoren (p-Typ, positiv), wodurch sich die elektrische Leitfähigkeit des Siliziums erhöht. [RS 25]

11.2.1. N-Kanal-MOS-FET

- Geläufige Bezeichnungen: N-Channel-MOSFET, NMOS, NMOSFET
- Dotierung: Elektronen-Donatoren – das Fremdatom besitzt ein zusätzliches Elektron, das als Ladungsträger fungiert
- Majoritätsladungsträger: Elektronen
- Funktionsweise: Der Transistor leitet bei positiver Spannung am Gate-Anschluss
- Schaltverhalten: Das Source-Potential ist niedriger als das Drain-Potential

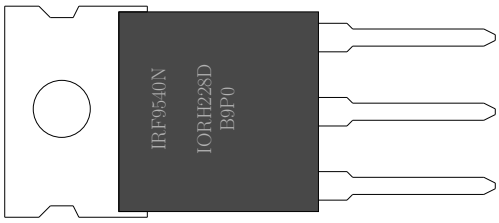
11.2.2. P-Kanal-MOS-FET

- Geläufige Bezeichnungen: P-Channel-MOSFET, PMOS, PMOSFET
- Dotierung: Elektronen-Akzeptoren – dem Fremdatom fehlt ein Elektron, es entsteht ein sogenanntes „Loch“
- Majoritätsladungsträger: Löcher (Defektelektronen)
- Funktionsweise: Der Transistor leitet bei negativer Gate-Spannung relativ zur Source
- Schaltverhalten: Das Source-Potential ist höher als das Drain-Potential

11.3. MOS-FET IRF9540N

Der COM-MOSFET ist ein p-Kanal MOSFET, der für die Steuerung von höheren Spannungen entwickelt wurde. Er eignet sich besonders für Anwendungen, bei denen eine effektive Spannung mit Hilfe von Pulsweitenmodulation (PWM) gesenkt werden soll, wie z. B. beim Dimmen von LED-Beleuchtungen.[Rei25a] 11.1

Figure 11.1.: Skizze des MOS-FETs



11.4. Spezifikation

Hauptmerkmale [Rei25a]

- Modell: IRF9540N
- Farbe der Platine: Schwarz

- Kompatibilität: Arduino, Raspberry Pi, etc.

- Besonderheiten: Der MOSFET bezieht Spannung auch über den SBC, wenn keine Source-Spannung verfügbar ist.

- Effektivspannung lässt sich mit PWM senken.

- Abmessungen: 40 mm x 22 mm x 20 mm

- Maximale Eingangsspannung: 36 V

- Maximale Eingangsstrom: 2 A

Steuerung und Anschlüsse [Rei25a]

- Steuerspannung: Mindestspannung 2 V

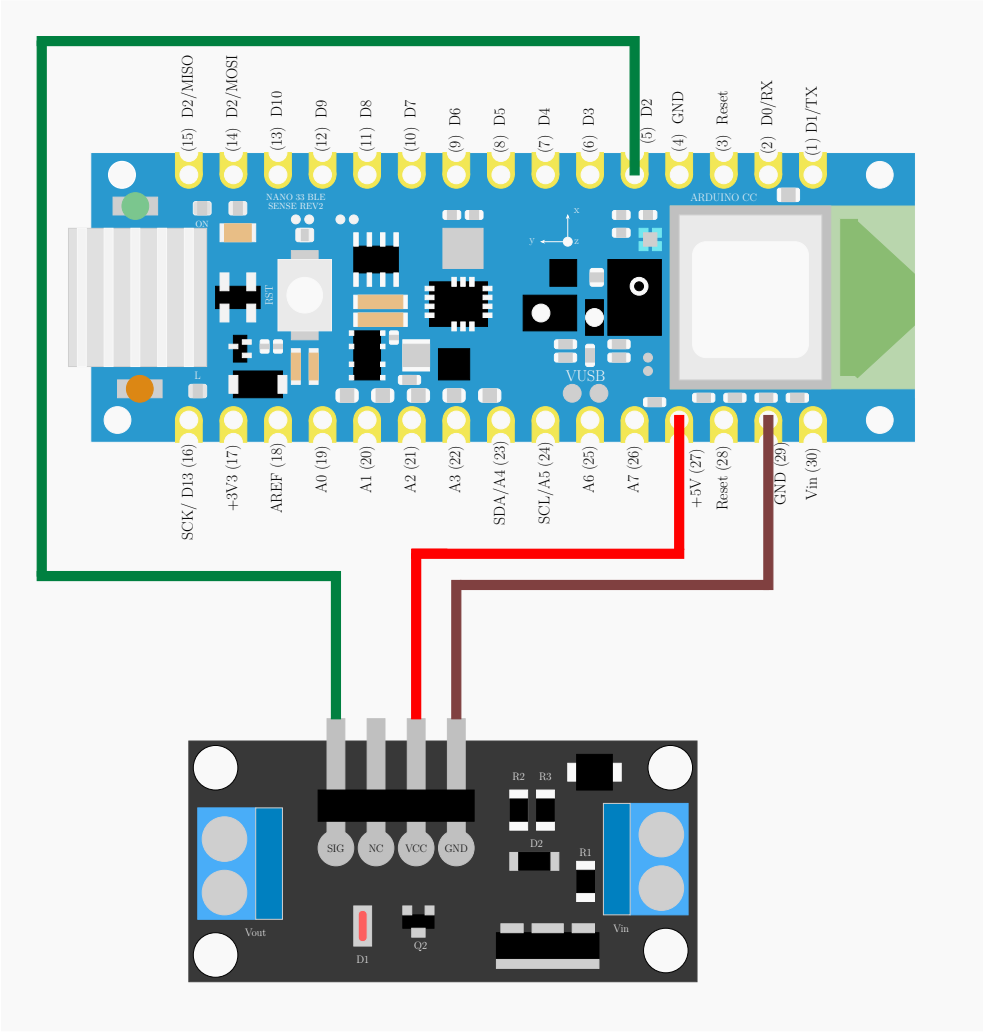
- Anschlüsse:
  - 2x 2 Pin Anschlussblöcke
  - 4 Pin female JST-Buchse

- Pinbelegung gemäß der Tabelle 11.1 11.2

Table 11.1.: Pinbelegung für das MOS-FET [Rei22]

Arduino	MOS-FET
GND	GND (Pin1)
5V	VCC (Pin2)
NC	NC (Pin3)
Digital Pin 2	Signal (Pin4)

Figure 11.2.: Pinbelegung des MOS-FETs



11.5. Einfacher Code

Dies ist ein einfacher Code zur Ansteuerung eines MOSFETs. Wird der Signal-Pin auf High gesetzt, aktiviert sich der Ausgang. Im folgenden Codebeispiel wird der Ausgang alle 15 Sekunden für eine Dauer von 10 Sekunden eingeschaltet.[Rei22]

Listing 11.1.: Einfacher Code zum Testen des MOS-FETs

```
from time import sleep
import RPi.GPIO as GPIO

# Set the pin numbering mode to BOARD (uses physical pin numbers)
GPIO.setmode(GPIO.BOARD)

# Define the signal pin number
Signal_Pin = 16

# Set the signal pin as an output
GPIO.setup(Signal_Pin, GPIO.OUT)

try:
    # Infinite loop to toggle the output
    while True:
        # Turn the output ON (set pin HIGH)
        GPIO.output(Signal_Pin, GPIO.HIGH)
        print("Output activated")
        sleep(10)  # Keep output on for 10 seconds

        # Turn the output OFF (set pin LOW)
        GPIO.output(Signal_Pin, GPIO.LOW)
        print("Output deactivated")
        sleep(5)  # Wait 5 seconds before turning it on again

except KeyboardInterrupt:
    # Clean up GPIO settings when the program is interrupted (e.g., with
      Ctrl+C)
    GPIO.cleanup()
```

../../Code/Arduino/MOSFET/simpleMOSFET.ino

11.6. Tests

Um einen MOS-FET zu testen, kann man eine einfache Widerstandsmessung mit einem Multimeter durchführen. Zuerst stellt man das Multimeter auf Widerstandsmessung und verbindet die Messspitzen mit den Drain- und Source-Anschlüssen des MOS-FETs. Ist keine Spannung am Gate angelegt, sollte der Widerstand zwischen Drain und Source hoch sein, was bedeutet, dass der MOS-FET „aus“ ist und keinen Strom fließen lässt. Wird jedoch eine Spannung (z. B. 9V) zwischen Gate und Source angelegt, sollte der Widerstand zwischen Drain und Source stark sinken, da der MOS-FET nun „ein“ ist und den Stromfluss ermöglicht. Diese einfache Prüfung bestätigt, ob der MOS-FET ordnungsgemäß funktioniert.

11.7. Einfache Anwendung

In dieser einfachen Schaltung wird ein MOS-FET (z.B. IRF9540N) verwendet, um eine LED ein- und auszuschalten. Der Gate-Anschlus des MOS-FETs wird direkt von einem digitalen Ausgang eines Mikrocontrollers (z.B Arduino) angesteuert. Wenn das Gate auf einen bestimmten Pegel geschaltet wird, leitet der MOS-FET und die LED leuchtet. Wird das Gate-Signal entfernt oder umgekehrt, unterbricht der MOS-FET den Stromfluss und die LED geht aus. Diese Anwendung zeigt das grundlegende Prinzip: ein kleiner Steuerstrom am Gate kann eine größere Last im Drain-Source-Zweig ein- oder ausschalten.

Listing 11.2.: Einfache Anwendung zum Verwenden des MOS-FETs

```
/**
 * @brief Simple MOSFET control example using an LED
 * Turns an LED on and off using a MOSFET and Arduino pin.
 */

#define MOSFET_GATE_PIN 3

void setup() {
  pinMode(MOSFET_GATE_PIN, OUTPUT);
}

void loop() {
  digitalWrite(MOSFET_GATE_PIN, LOW);   // For P-Kanal MOSFET: Turn ON
  delay(1000);                          // LED ON for 1 second

  digitalWrite(MOSFET_GATE_PIN, HIGH);  // Turn OFF
  delay(1000);                          // LED OFF for 1 second
}
```

../../Code/Arduino/MOSFET/testMOSFET.ino

12. Externe LED

12.1. Einleitung

Leuchtdioden, kurz LEDs (Light Emitting Diodes), sind aus der heutigen Elektronik nicht mehr wegzudenken. Sie bieten eine energieeffiziente, langlebige und kompakte Lösung zur Lichterzeugung – sei es in Haushaltsgeräten, Automobiltechnik, Signalanlagen oder modernen Beleuchtungssystemen. Aufgrund ihrer hohen Schaltgeschwindigkeit und Robustheit sind LEDs auch in der Kommunikationstechnik und im Maschinenbau weit verbreitet. [HI18] [Her+24]

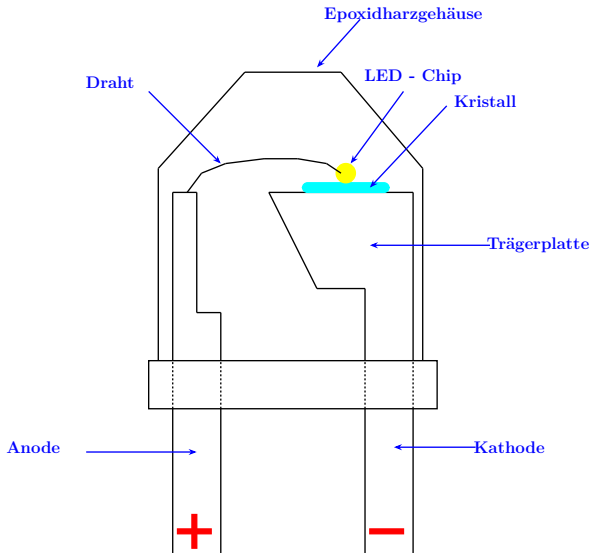
12.2. Grundlegende Informationen

Eine LED ist eine spezielle Form der Diode, die bei Durchfluss eines elektrischen Stroms Licht emittiert. Im Gegensatz zu herkömmlichen Glühlampen erzeugen LEDs Licht nicht durch Erhitzen eines Drahts, sondern durch sogenannte Elektrolumineszenz: Elektronen und „Löcher“rekombinieren in einem Halbleitermaterial (meist Gallium- oder Siliciumverbindungen). Rekombinieren bedeutet, dass ein freies Elektron in ein Loch fällt und dessen Energieplatz im Kristallgitter besetzt. Dabei verliert das Elektron Energie – diese Energie wird nicht als Wärme, sondern als Licht (Photon) abgegeben. Die Menge an Energie, die beim Rekombinieren freigesetzt wird, entspricht der Bandlücke des Halbleitermaterials – also dem Unterschied zwischen dem Energiezustand des Elektrons und dem des Lochs. Diese Energie wird als Licht einer bestimmten Farbe (also mit bestimmter Wellenlänge) abgestrahlt. Eine große Bandlücke bedeutet kurzwelliges Licht (Blau, UV) und eine kurze Bandlücke bedeutet langwelliges Licht (Rot). [HI18] [Her+24] [Fal25]

Grundlegend ist eine LED wie in 12.1 mit den folgenden Komponenten aufgebaut:

- Anode und Kathode (positive und negative Anschlüsse)
- LED - Chip, der das Licht erzeugt
- Halbleiter-Kristall (Silicium/Germanium)
- Epoxidharzgehäuse: schützt das Bauteil und lässt das Licht ausstrahlen
- Gegebenenfalls Vorlaufwiderstand

Figure 12.1.: Aufbau einer LED



12.3. LED CML Signalleuchte, blau, ultrahell, 12 VDC

Die Signalleuchte von CML ist eine ultrahelle blaue LED mit einem Durchmesser von 8 mm. Sie eignet sich besonders für Anwendungen, bei denen eine auffällige visuelle Anzeige erforderlich ist. [Rei25c] [CML11]

12.4. Spezifikation

- Farbe: Blau
- Lichtstärke: 600 mcd
- Versorgungsspannung: 12 V AC/DC
- Einbaugröße: 8 mm Durchmesser, passend für Standard-Bohrungen in Frontplatten oder Gehäusen
- Anschluss: FASTON-Anschlüsse oder Lötanschlüsse für einfache Integration
- Besonderheiten: Integrierter Vorwiderstand, der den direkten Anschluss an 12 V ermöglicht, ohne dass zusätzliche Komponenten erforderlich sind

[Rei25c] [CML11]

12.5. Einfacher Code

Ein einfacher Code wäre die Ansteuerung einer externen LED.12.1

Listing 12.1.: Einfacher Code zur Ansteuerung einer externen LED

```
/**
 * @file ansteuerungLED.ino
 * @brief Simple LED control with Arduino
 *
 * This sketch turns an LED on and off at 1-second intervals.
 * It demonstrates basic digital output control using the Arduino
   framework.
 */

/**
 * @brief Pin number where the LED is connected
 */
const int ledPin = 13;  ///< LED connected to digital pin 13

/**
 * @brief Initializes the LED pin as an output
 */
void setup() {
  // Set the LED pin as output
  pinMode(ledPin, OUTPUT);
}

/**
 * @brief Main loop: turns the LED on and off with 1-second delay
 */
void loop() {
  digitalWrite(ledPin, HIGH);  // Turn the LED on
  delay(1000);                 // Wait for 1 second
  digitalWrite(ledPin, LOW);   // Turn the LED off
  delay(1000);                 // Wait for 1 second
}
```

../../Code/Arduino/Externe LED/ansteuerungLED/ansteuerungLED.ino

12.6. Tests

Die LED ist leicht zu testen indem man sie an eine 12 V Stromversorgung anschließt, ein Vorwiderstand ist bei dieser spezifischen LED nicht nötig, da dieser bereits integriert ist.

12.7. Einfache Anwendung

Als einfache Anwendung eignet sich eine Blinkfrequenz-Schaltung. Dieser Code 12.2 lässt eine LED drei mal mit 1 Hz blinken, dann fünf mal bei 2 Hz und dann zehn mal mit 5 Hz.

Listing 12.2.: Einfacher Code zum Blinken einer externen LED

```
/**
 * @file blinkenLED.ino
 * @brief LED blinking with multiple frequencies
 *
 * This sketch blinks an LED at different frequencies sequentially.
 * It demonstrates control of delay timing to create various blink
    patterns.
 */

/**
 * @brief Pin number where the LED is connected
 */
const int ledPin = 13;  ///< LED connected to digital pin 13

/**
 * @brief Initializes the LED pin as an output
 */
void setup() {
  // Set the LED pin as output
  pinMode(ledPin, OUTPUT);
}

/**
 * @brief Blinks the LED with the specified on/off delay and repetitions
 * @param delayTime Delay time in milliseconds between on and off states
 * @param repetitions Number of times the LED should blink
 */
void blinkPattern(int delayTime, int repetitions) {
  for (int i = 0; i < repetitions; i++) {
    digitalWrite(ledPin, HIGH);  // Turn the LED on
    delay(delayTime);            // Wait
    digitalWrite(ledPin, LOW);   // Turn the LED off
    delay(delayTime);            // Wait
  }
}

/**
 * @brief Main loop: executes multiple blink patterns in sequence
 */
void loop() {
  blinkPattern(1000, 3);  // Blink slowly (1 Hz) three times
  blinkPattern(500, 5);   // Blink at medium speed (2 Hz) five times
  blinkPattern(200, 10);  // Blink fast (5 Hz) ten times
  delay(2000);            // Wait before repeating the sequence
}
```

../../Code/Arduino/Externe LED/blinkenLED/blinkenLED.ino

12.8. Further Readings

Haeberle and Isele. Tabellenbuch Elektrotechnik. 28th ed. Europa Lehrmit- tel, 2018. isbn: 978-3-808-53490-8.[HI18]

- Ein tolles Tabellenbuch in dem viel Grundlagenwissen vermittelt wird. Zudem sind die einfache Skizzen und Erklärungen hervorzuheben.

Falcon Illumination. Wie funktioniert eine LED? [pdf]. 2025. URL: `https://www.falconillumination.de/de/wissenswertes/wie-funktioniert-eine-led.html` [Fal25]

- Sehr spannende Website mit einfachen Erklärungen rund um das Thema Beleuchtungstechnik.

Part III.

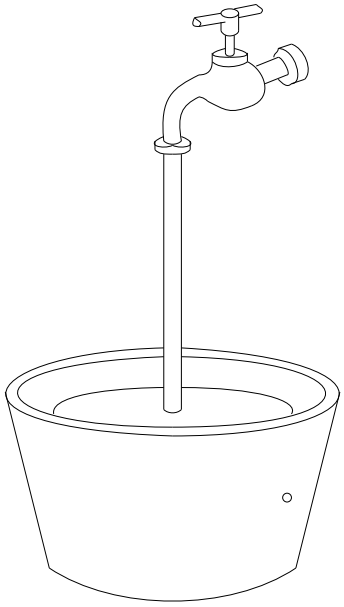
Applikation: Magic Faucet Fountain

13. Projekt: Magic Faucet Fountain

13.1. Beschreibung

Das Projekt Magic Faucen Fountain beschäftigt sich mit der Umsetzung eines Brunnens, der die optische Täuschung eines schwebenden Wasserhahn erzeugt. Dabei scheint es, als würde Wasser kontinuierlich aus einem frei schwebenden Hahn fließen - ganz ohne sichtbare Verbindung zur Wasserquelle 13.1.

Figure 13.1.: Skizze des Magic Faucet Fountain



Dieser Effekt wird durch ein transparentes Acrylrohr realisiert, das sowohl den Wasserfluss leitet als auch die tragende Struktur für den Hahn darstellt. Da das Rohr hinter dem gleichmäßig fließenden Wasser optisch kaum wahrnehmbar ist, entsteht der Eindruck eines schwebenden Hahns.
Der Wasserstrom wird über eine kleine elektrische Pumpe erzeugt, die über ein MOS-FET angesteuert wird. Der MOS-FET fungiert als elektronischer Schalter und wird von einem digitalen Ausgang des Arduino-Boards gesteuert. Der Arduino ist über Bluetooth mit einer mobilen App verbunden. Über die App kann der Benutzer den Brunnen bequem ein- und ausschalten.
Ein weiterer Bestandteil des Systems ist ein Ultraschallsensor, der den Wasserstand im Vorratsbehälter misst. Der aktuelle Füllstand wird ebenfalls in der App angezeigt, wodurch der Benutzer rechtzeitig erkennen kann, wann Wasser nachgefüllt werden muss.

13.2. Aufbau

Die mechanische Struktur des Magic Faucet Fountain basiert auf einem dekorativen Blumentopf, der gleichzeitig als Wasserreservoir und Trägerelement für den sicht-

baren Aufbau dient. Der Blumentopf wird durch einen funktional gestalteten Deckel verschlossen, der sowohl mechanische als auch gestalterische Funktionen erfüllt.
Zentral im Deckel befindet sich eine Durchgangsbohrung, durch die ein transparentes Acrylrohr senkrecht nach oben geführt wird. Am oberen Ende des Rohrs ist ein dekorativer Wasserhahn angebracht. Direkt unterhalb dieses Hahns befinden sich feine Austrittsbohrungen, durch die Wasser kontrolliert herausfließt. Aufgrund der Transparenz des Rohrs und des gleichmäßigen Wasserstroms entsteht die Illusion eines frei schwebenden, endlos fließenden Wasserhahns.
Seitlich im Deckel befinden sich zusätzliche Rücklaufbohrungen, über die das Wasser zurück in den Blumentopf gelangt. Eine im Inneren platzierte Wasserpumpe sorgt für den geschlossenen Wasserkreislauf, indem sie das Wasser wieder nach oben in das Acrylrohr befördert.
An der Unterseite des Deckels ist eine Gewindeaufnahme integriert, an der ein senkrecht montiertes Führungsrohr angebracht ist. In diesem Rohr befindet sich ein frei beweglicher Schwimmer, der den Wasserstand abbildet. Das Rohr ist unten mit einem fest montierten Deckel verschlossen, in den ein Gitter direkt eingearbeitet ist. Dieses verhindert das Herausfallen des Schwimmers, ermöglicht aber den Wassereintritt. Über der Rohröffnung ist ein Ultraschallsensor montiert, der den Abstand zum Schwimmer misst und so den Wasserstand erkennt.
Die elektronischen und steuerungstechnischen Komponenten wie Arduino, MOS-FET, Spannungsversorgung, Bluetooth-Modul sowie die Status-LEDs und der Kippschalter befinden sich in einem separaten Gehäuse, das als elektronischer Kasten (E-Kasten) ausgeführt ist. Dieser Kasten schützt die Elektronik vor Feuchtigkeit und erlaubt einen strukturierten, servicefreundlichen Aufbau. Die Leitungen zur Pumpe, zum Ultraschallsensor und zu den LEDs werden durch entsprechende Kabeldurchführungen zwischen dem E-Kasten und der Hauptbaugruppe verbunden.
Zur optischen Aufwertung kann die Oberseite des Deckels mit dekorativen Steinen oder anderen Gestaltungselementen versehen werden, wodurch die technische Funktion harmonisch in ein ästhetisches Gesamtbild eingebettet wird.

13.2.1. Abmaß

Da der endgültige Aufbau des Projekts „Magic Faucet Fountain“ zum Zeitpunkt der Dokumentation noch nicht erfolgt ist, können die Abmessungen lediglich auf Basis der entworfenen Komponenten und Gehäusekonzepte geschätzt werden. Die Steuerungseinheit inklusive Arduino, MOS-FET-Schaltung und Stromversorgung soll in einem kompakten Gehäuse untergebracht werden. Die Wasserinstallation mit Pumpe und transparentem Rohrsystem wird so dimensioniert, dass sie sich auf einer Fläche von ca. 40 cm × 40 cm bei einer Höhe von maximal 60 cm aufbauen lässt. Genauere Angaben erfolgen nach Fertigstellung des physischen Prototyps.

13.3. Hardware-Schnittstellen

Die Magic Faucet Fountain besteht aus verschiedenen Hardware-Komponenten, welche durch physische Verbindungen miteinander kommunizieren und Daten austauschen. In diesem Kapitel sollen diese Schnittstellen mit Schaltplänen veranschaulicht mittels dieser erklärt werden.

13.3.1. Schaltpläne

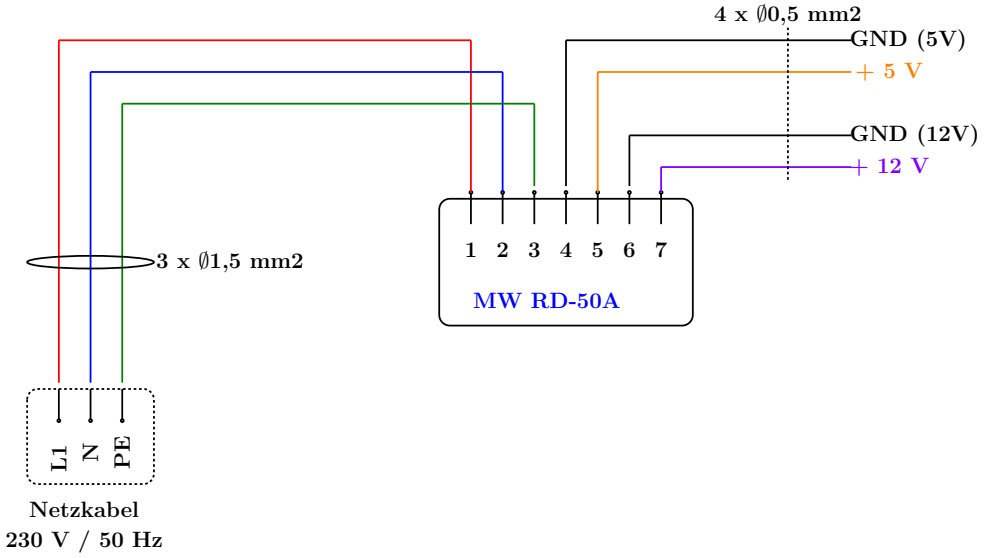
Versorgungsspannungen 5 V und 12 V

Die im Schaltplan in Abbildung 13.2 dargestellten Komponenten umfassen den Netzanschluss (230 V AC / 50 Hz) sowie das Schaltnetzteil vom Typ MW RD-50A. Dieses wandelt die Netzspannung in geregelte Gleichspannungen von 5 V und 12 V DC um,

die für den Betrieb der Steuereinheit und der Aktoren erforderlich sind. Der Anschluss erfolgt über:

- 3-adriges Netzkabel (1,5 mm^2)
- 4-adriges Verbindungskabel (0,5 mm^2) für:
  - +5 V (z. B. Arduino, BT-Modul)
  - GND (5 V)
  - +12 V (z. B. Pumpe)
  - GND (12 V)

Figure 13.2.: Versorgungsschaltplan



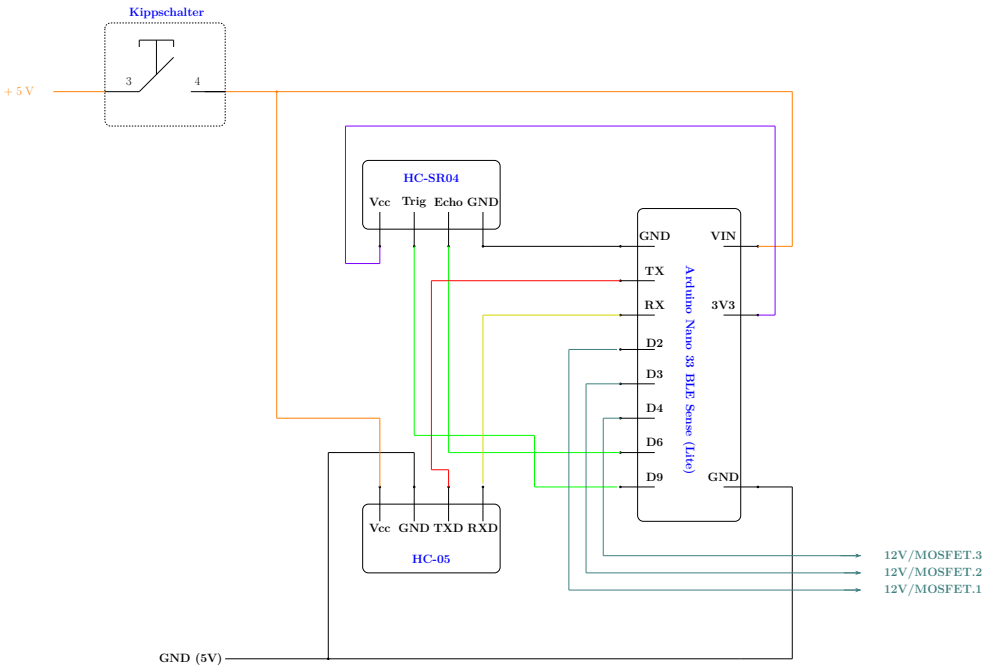
5 V Netz mit Arduino und BT-Modul

Im Schaltplan 13.3 sind der Arduino Nano, das HC-05 Bluetooth-Modul und die Stromversorgung durch das Netzteil als Komponenten zu entnehmen. Es lassen sich folgende Schnittstellen aus der Schaltung herleiten:

- UART-Schnittstelle für die Bluetooth-Kommunikation (siehe TXD/RXD Verbindung zwischen Arduino und HC-05
- Digitale Ausgänge D2, D3, D4 zur Steuerung der Gate-Anschlüsse der MOS-FET-Schaltung für die Pumpe und LEDs
- Spannungsversorgung 5 V / GND

Der Arduino verarbeitet die über das Bluetooth-Modul eingehenden Steuerbefehle und schaltet gezielt die Ausgänge zur Ansteuerung der Aktoren. Über den HC-05 erfolgt die drahtlose Verbindung mit einem Smartphone.

Figure 13.3.: Schaltplan: Arduino-basierte Steuerungseinheit mit Bluetooth-Kommunikation



5/12 V Netz mit Aktoren

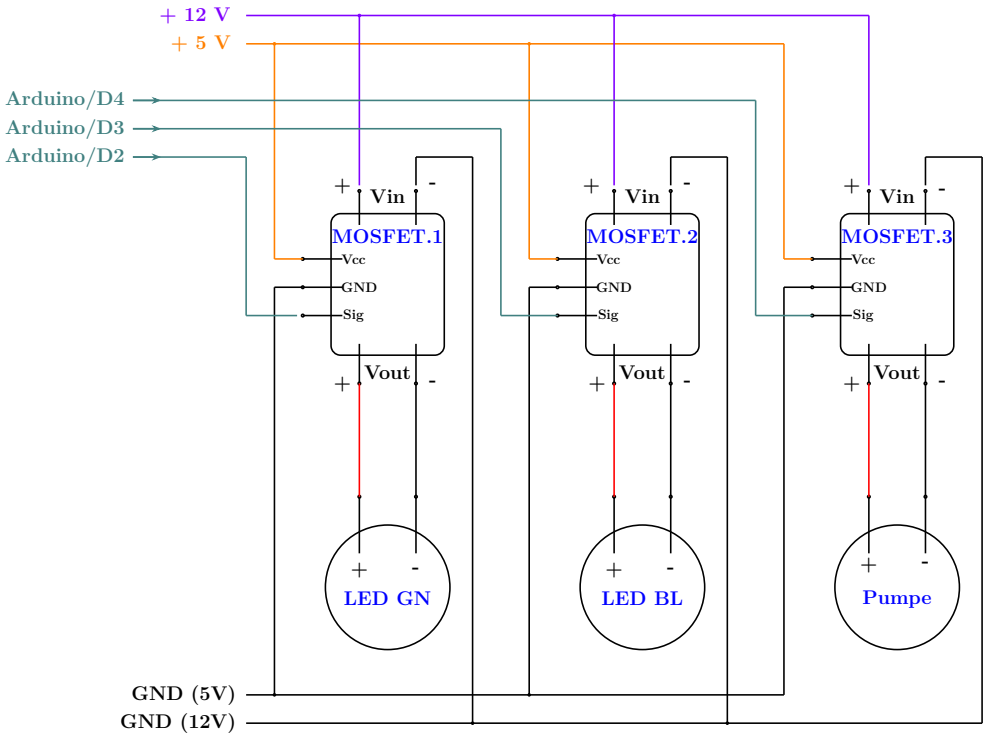
Komponenten dieser Schaltung sind: 3x N-Kanal-MOS-FETs, LED-Leuchten (grün und blau) und die Wasserpumpe. Dem Aufbau der Schaltung ist zu entnehmen, dass jeder MOS-FET als Schalter arbeitet, der über den Arduino gesteuert wird. Die genaue Ansteuerung der Komponenten durch den Arduino wird in Tabelle 13.1 dargestellt.

Table 13.1.: Ansteuerung der Komponenten durch den Arduino

MOS-FET	Gesteuertes Element	Arduino Pin	Versorgungsspannung
1	LED grün	D4	5 V
2	LED blau	D3	5 V
3	Wasserpumpe	D2	12 V

Die +Vin-Anschlüsse der Aktoren erhalten jeweils die notwendige Betriebsspannung direkt vom Netzteil, während die Schaltung über den MOS-FET den Stromfluss zum Masseanschluss (GND) kontrolliert. Eine detaillierte Darstellung der Schaltung ist in Abbildung 13.4 zu finden.

Figure 13.4.: 5/12 V Netz mit Aktoren



13.4. Software-Schnittstellen

In dem Projekt Magic Faucet Fountain, bei dem ein Mikrokontroller bzw. der Arduino und ein Bluetooth-Modul (hier: HC-05) verwendet werden, beziehen sich Software-Schnittstellen auf die programmatische Kommunikation und Steuerung zwischen den Systemkomponenten.

Bluetooth-Kommunikationsschnittstelle

Die an der Bluetooth-Kommunikation beteiligten Komponenten sind der Arduino, HC-05 Bluetooth-Modul und das Smartphone mit der Bluetooth-App. Die serielle Kommunikation über UART (TX/RX) stellt die Hauptschnittstelle für die Steuerung dar. Eine Bluetooth-App sendet ASCII-Befehle oder einfache Steuersignale an den Arduino. Dieser wertet die empfangenen Daten aus und aktiviert über seine digitalen Pins die entsprechenden MOS-FET-Schaltungen.

Arduino-Programmierschnittstelle

Der Arduino Mikrokontroller wird mit einem Arduino Sketch (Arduino DIE) programmiert, der folgende Aufgaben übernimmt:

- Initialisierung der Pins
- Einrichtung der seriellen Schnittstelle (Serial.begin(9600))
- Einlesen und Interpretieren von Bluetooth-Befehlen
- Setzen der Ausgänge zur Ansteuerung der Aktoren

Anwendungsprogrammierschnittstelle auf Befehlsebene

Die Bluetooth-App kommuniziert mit dem Arduino über eine einfache Befehlssprache, z.B.:

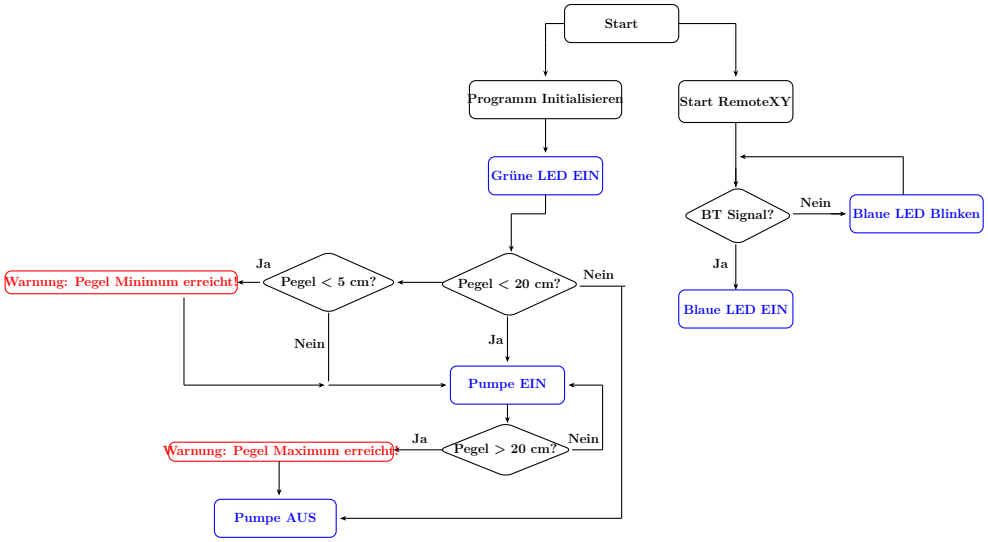
- „On1“: LED ein
- „Off3“: Pumpe aus

Der Arduino-Sketch interpretiert diese Befehle als Teil eines einfachen protokollbasierten Interfaces. Diese Art der Schnittstelle ist textbasiert.

13.4.1. Ablaufdiagramm der Anwendung

Zur Veranschaulichung des Zusammenspiels der beschriebenen Software-Schnittstellen dient das in Abbildung 13.5 dargestellte Ablaufdiagramm. Es zeigt den strukturierten Programmablauf vom Start der Anwendung über die Initialisierung und Bluetooth-Kommunikation bis hin zur Ansteuerung der LEDs und der Wasserpumpe. Zudem wird der Umgang mit kritischen Zuständen wie zu hohen oder zu niedrigen Wasserständen abgebildet.

Figure 13.5.: Ablaufdiagramm



13.4.2. Code der Anwendung

```
/**
 * @file ablaufApplikation.ino
 * @brief Automated flowerpot with Bluetooth status, pump control, and
    water level measurement.
 */

#include <SoftwareSerial.h>
#include <RemoteXY.h>

// RemoteXY Bluetooth (HC-05) via SoftwareSerial
#define REMOTEXY_SERIAL_RX 10
#define REMOTEXY_SERIAL_TX 11
#define REMOTEXY_SERIAL_SPEED 9600

#pragma pack(push, 1)
uint8_t RemoteXY_CONF[] =
{ 255,1,0,0,0,60,0,13,13,1,
 2,0,7,7,18,8,2,26,31,80,
 117,109,112,101,0,1,0,30,7,
 40,10,2,31,65,110,0,79,117,
 115,0 };

struct {
  uint8_t pumpSwitch;        ///< 0 = Off, 1 = On
  char statusText[128];      ///< Text display for warnings
  uint8_t connect_flag;      ///< Bluetooth connection status
} RemoteXY;
#pragma pack(pop)

// Pin definitions
const int PIN_LED_GREEN = 2;
const int PIN_LED_BLUE = 3;
const int PIN_PUMP = 4;
const int PIN_TRIGGER = 9;
const int PIN_ECHO = 6;

const int MAX_WATER_LEVEL = 20;
const int MIN_WATER_LEVEL = 5;

unsigned long lastBlinkTime = 0;
bool blueLedState = false;

/**
 * @brief Initializes pins and RemoteXY
 */
void setup() {
  pinMode(PIN_LED_GREEN, OUTPUT);
  pinMode(PIN_LED_BLUE, OUTPUT);
  pinMode(PIN_PUMP, OUTPUT);
  pinMode(PIN_TRIGGER, OUTPUT);
  pinMode(PIN_ECHO, INPUT);

  digitalWrite(PIN_LED_GREEN, HIGH); // Green LED always on

  RemoteXY_Init();
}

/**
 * @brief Main loop: water level measurement, display update, Bluetooth
    status, and pump control
 */
void loop() {
  RemoteXY_Handler();
  updateBluetoothStatusLED();

  int distance = measureWaterLevel();

  // Basic warning message
  if (distance > MAX_WATER_LEVEL) {
    strcpy(RemoteXY.statusText, "WARNING: No water left!");
  }
  else if (distance < MIN_WATER_LEVEL) {
    strcpy(RemoteXY.statusText, "WARNING: Low level!");
    if (RemoteXY.pumpSwitch) {
      strcat(RemoteXY.statusText, " - Pump deactivated!");
    }
  }
  else {
```

13.5. Installation

Die Installation des Magic Faucet Fountain gliedert sich in zwei Hauptbereiche: den mechanischen Aufbau der Baugruppe und die elektrische Inbetriebnahme der Steuerungskomponenten. Im Folgenden werden beide Teile Schritt für Schritt erläutert.

13.5.1. Mechanischer Aufbau

- Der Blumentopf wird als zentrales Bauteil aufgestellt und mit einem speziell angefertigten Deckel verschlossen. Dieser beinhaltet:
  - Eine zentrale Öffnung zur Durchführung des Acrylrohrs, das den Wasserfluss führt.
  - Mehrere Rücklaufbohrungen, durch die Wasser in den Topf zurückfließt.
  - Eine Gewindeaufnahme auf der Unterseite zur Befestigung des Führungsrohrs für den Schwimmer.
- Das transparente Acrylrohr wird vertikal durch den Deckel geführt und oben mit einem dekorativen Wasserhahn verbunden. Kleine Bohrungen im Rohr direkt unter dem Hahn ermöglichen den sichtbaren Wasseraustritt.
- Das Führungsrohr wird senkrecht unter dem Deckel montiert. Im Inneren befindet sich ein Schwimmer, der sich mit dem Wasserstand bewegt.
- Der untere Abschluss des Führungsrohrs besteht aus einem fest verschlossenen Deckel mit integriertem Gitter, das den Schwimmer sichert und den Wassereintritt erlaubt.
- Der Ultraschallsensor wird oberhalb des Führungsrohrs an der Aufnahme befestigt und misst den Abstand zum Schwimmer zur Bestimmung des Füllstands.
- Auf dem Deckel können zur Gestaltung Dekosteine oder andere Elemente platziert werden.

13.5.2. Elektrische Installation

Alle elektronischen Komponenten sind im separaten E-Kasten untergebracht, der außerhalb des Blumentopfs montiert wird und Feuchtigkeitsschutz sowie einfache Wartung ermöglicht.
Folgende Komponenten sind dort integriert:

- Arduino-Mikrocontroller, als zentrale Steuereinheit.
- MOS-FET-Schaltkreis zur Ansteuerung der Tauchpumpe, die das Wasser durch das Acrylrohr fördert.
- Ultraschallsensor, über digitale Pins mit dem Arduino verbunden.
- Bluetooth-Modul, über serielle Schnittstelle angebunden.
- Zwei Status-LEDs:
  - Grüne LED: zeigt den Betriebszustand an – leuchtet, wenn der Kippschalter eingeschaltet ist.
  - Blaue LED: zeigt den Bluetooth-Zustand – blinkt bei Verbindungssuche, leuchtet bei bestehender Verbindung.
- Kippschalter als Hauptschalter für das gesamte System.
- Stromversorgung über externes Netzteil, angeschlossen an den E-Kasten.

- Alle Verbindungen zur Pumpe, zu den Sensoren und zu den LEDs sind über robuste Steckverbindungen aus dem E-Kasten herausgeführt.

13.5.3. Software und App-Verbindung

- Nach dem Einschalten über den Kippschalter (→ grüne LED leuchtet), startet der Arduino.
- Die Bluetooth-Verbindung wird automatisch aktiviert (→ blaue LED blinkt).
- Über eine passende Smartphone-App kann:
  - Die Pumpe ein- und ausgeschaltet werden.
  - Der aktuelle Wasserstand basierend auf den Sensordaten angezeigt werden.
  - Der Bluetooth-Status überwacht werden.

13.5.4. Software und App-Verbindung

1. Wasser einfüllen: Der Topf wird so weit befüllt, dass die Pumpe vollständig untergetaucht ist.
2. System einschalten: Kippschalter betätigen → grüne LED zeigt Betriebsbereitschaft.
3. Bluetooth koppeln: Verbindung mit der App herstellen → blaue LED bleibt dauerhaft an.
4. Funktionstest: Pumpe über App starten, Wasserfluss beobachten.

13.6. Konfiguration

Die Konfiguration des Magic Faucet Fountain umfasst die Inbetriebnahme und Abstimmung der elektronischen Komponenten sowie die Kopplung mit der mobilen App. Ziel ist es, die Steuerung des Brunnens per Bluetooth zuverlässig zu ermöglichen und eine präzise Füllstandsanzeige über den Ultraschallsensor bereitzustellen.

1. Elektrische Inbetriebnahme
   - Vor dem Einschalten müssen alle elektrischen Verbindungen überprüft werden: Die Pumpe, der Ultraschallsensor, die Status-LEDs, der MOS-FET, der Kippschalter und das Bluetooth-Modul müssen korrekt mit dem Arduino verbunden sein.
   - Die Stromversorgung wird über den Kippschalter aktiviert. Beim Einschalten leuchtet die grüne Betriebs-LED zur Bestätigung.
2. Bluetooth-Konfiguration
   - Nach dem Einschalten beginnt die blaue LED zu blinken, was signalisiert, dass das Bluetooth-Modul aktiv ist und auf eine Verbindung wartet.
   - Über die App wird das Bluetooth-Modul des Brunnens gesucht und gekoppelt. Nach erfolgreicher Verbindung wechselt die blaue LED von Blinken auf Dauerlicht.
   - Die App ermöglicht das manuelle Ein- und Ausschalten der Pumpe sowie die Anzeige des aktuellen Wasserstands.
3. Füllstanderfassung

- Der Ultraschallsensor misst kontinuierlich den Abstand zum Schwimmer im Führungsrohr. Die Werte werden vom Arduino verarbeitet und über Bluetooth an die App gesendet.
- In der App wird der Wasserstand durch einen Ladebalken visualisiert und zusätzlich als Zahlenwert in Millilitern angegeben.

4. Software-Einstellungen
   - Die Arduino-Firmware enthält die Steuerlogik für:
     - Die Pumpensteuerung (an/aus via MOS-FET).
     - Die Auswertung der Ultraschallsensordaten.
     - Die Ansteuerung der Status-LEDs.
5. Systemtest
   - Nach der vollständigen Konfiguration sollte ein Testlauf durchgeführt werden, bei dem:
     - Die Bluetooth-Verbindung getestet wird.
     - Der Wasserstand korrekt in der App angezeigt wird.
     - Die Pumpe zuverlässig über die App geschaltet werden kann.
     - Die LEDs den korrekten Verbindungs- und Betriebsstatus anzeigen.
   - Über die App wird das Bluetooth-Modul des Brunnens gesucht und gekoppelt. Nach erfolgreicher Verbindung wechselt die blaue LED von Blinken auf Dauerlicht.
   - Die App ermöglicht das manuelle Ein- und Ausschalten der Pumpe sowie die Anzeige des aktuellen Wasserstands.

13.7. Einschränkungen

Da das Projekt sich noch in der finalen Planungs- und Dokumentationsphase befindet, konnten einige Einschränkungen nur theoretisch erkannt werden:

- Praxistests fehlen bislang, weshalb reale Probleme bei der Wasserführung, Abdichtung oder elektrischen Sicherheit noch nicht abschließend beurteilt werden können.
- Die Stromversorgung muss bei der Kombination von 5 V- und 12 V-Komponenten mit Bedacht gewählt werden, um Störungen oder Überlastungen zu vermeiden.
- Die Bluetooth-Kommunikation hängt von der Stabilität der Verbindung und der verwendeten App ab, mögliche Verbindungsabbrüche sind noch nicht erprobt.
- Platzbedarf und Montage könnten im späteren Aufbau auf bisher nicht eingeplante bauliche Herausforderungen stoßen.

14. Schlussfolgerungen

14.1. Zusammenfassung der Projektergebnisse

Im Verlauf des Projekts wurde ein umfassendes Konzept für die Umsetzung des „Magic Faucet Fountain“ erarbeitet. Es umfasst die Auswahl geeigneter Aktoren und Sensoren, die Steuerung per Arduino sowie die Integration einer benutzerfreundlichen Bluetooth-Schnittstelle. Der gesamte Schaltungsaufbau wurde in der Theorie konzipiert und in der Dokumentation technisch detailliert beschrieben. Die Projektziele hinsichtlich Planung, Dokumentation und Softwareentwicklung konnten erfolgreich erreicht werden.

14.2. Reflexion über die Zielerreichung

Das angestrebte Ziel, ein steuerbares System zur Ansteuerung von Pumpe und LED-Komponenten zu konzipieren, wurde im Rahmen der Dokumentation vollständig vorbereitet. Der Aufbau des Prototyps konnte aus organisatorischen Gründen noch nicht umgesetzt werden, die dafür erforderlichen technischen Grundlagen sind jedoch detailliert erarbeitet. Die entwickelten Hardware- und Softwarekomponenten bilden eine tragfähige Basis für die spätere praktische Umsetzung. Die bisherigen Ergebnisse deuten darauf hin, dass die technische Realisierbarkeit gegeben ist.

14.3. Mögliche Weiterentwicklungen

Für künftige Ausbaustufen des Systems sind verschiedene funktionale Erweiterungen denkbar, unter anderem:

- Integration einer Echtzeit-Füllstandsanzeige über die App
- Automatisierte Steuerung mittels Timer oder Umweltsensorik (z. B. Helligkeit, Temperatur)
- Einbindung in WLAN-Netzwerke zur Steuerung über Smart-Home-Plattformen
- Erweiterung um Audiosignale zur Kombination von Licht-, Wasser- und Klangkomponenten

14.4. Zukünftige Anwendungen

Das entwickelte Steuerungskonzept lässt sich auf unterschiedliche Anwendungsszenarien übertragen, beispielsweise:

- Steuerungseinheiten für dekorative Wasserinstallationen
- Anschauungsobjekte im Bereich der technischen Bildung (z. B. Mikrocontroller, Regelungstechnik)
- Interaktive Systeme für Ausstellungen oder Präsentationsumgebungen
- Automatisierte Bewässerungslösungen in intelligenten Gartenanwendungen

Part IV.

Anhang

Literaturverzeichnis

[AG24] A. AG. *NewPing – Library Documentation*. 2024. URL: https://docs.arduino.cc/libraries/newping/.

[All24] M. Alles. *Einführung in die elektronische Schaltungstechnik*. 1st ed. Springer, 2024. ISBN: 978-3-662-69278-3.

[Ard21] Arduino, ed. *Arduino Nano 33 BLE Sense Rev2 with headers*. 2021. URL: https://store.arduino.cc/products/nano-33-ble-sense-rev2-with-headers.

[Ard21] Arduino, ed. *Check out Arduino Docs!* 2021. URL: https://www.arduino.cc/en/Guide.

[Ard25] Arduino AG. *Arduino IDE 2.0*. [pdf]. 2025. URL: https://docs.arduino.cc/software/ide/#ide-v2.

[CML11] CML. *Handbuch - CML LED-Signalleuchte*. [pdf]. 2011.

[Com04] Comet. *Handbuch - Comet Tauchpumpe 12 V Elegant*. [pdf]. 2004.

[Con25] Conrad. *Niedervolt Tauchpumpe Comet 12 V Elegant*. 2025. URL: https://www.conrad.de/de/p/comet-1300-01-00-niedervolt-tauchpumpe-600-l-h-5-5-m-1435461.html?refresh=true#productDownloads.

[FAD18] M. Fezari and A. Al Dahoud. "Integrated Development Environment "IDE" For Arduino". In: (Oct. 2018).

[FD22] P. Filipi and Dozie. *A Difference between Arduino Nano 33 BLE Sense vs. Arduino Nano 33 BLE Sense Lite*. [pdf]. 2022. URL: https://forum.arduino.cc/t/a-difference-between-a-n-33-ble-sense-vs-sense-lite/1030305.

[FLU25] FLUKE. *Anleitung zum Messen von Widerstand*. 2025. URL: https://www.fluke.com/de-de/mehr-erfahren/blog/digitalmultimeter/anleitung-zum-messen-von-widerstand.

[Fal25] Falcon Illumination. *Wie funktioniert eine LED?* [pdf]. 2025. URL: https://www.falconillumination.de/de/wissenswertes/wie-funktioniert-eine-led.html (visited on 05/03/2025).

[GG20] J. F. Gülich and J. F. Gülich. "Bauarten und Leistungsdaten". In: *Kreiselpumpen: Handbuch für Entwicklung, Anlagenplanung und Betrieb* (2020), pp. 45–82.

[HEG21] E. Hering, J. Endres, and J. Gutekunst. *Elektronik für Ingenieure und Naturwissenschaftler*. 8. [pdf]. Springer Vieweg, 2021. ISBN: 978-3-662-62697-9. DOI: 10.1007/978-3-662-62698-6.

[HI18] Haeberle and Isele. *Tabellenbuch Elektrotechnik*. 28th ed. Europa Lehrmittel, 2018. ISBN: 978-3-808-53490-8.

[HLG19] B. Heinrich, P. Linke, and M. Glöckler. *Grundlagen Automatisierung: Erfassen-Steuern-Regeln*. Springer-Verlag, 2019.

[HS09] S. Hesse and G. Schnell. *Sensoren für die Prozess-und Fabrikautomation*. Vol. 5. Springer, 2009.

[HS+12] E. Hering, G. Schönfelder, et al. *Sensoren in Wissenschaft und Technik*. Springer, 2012.

[HS23] E. Hering and G. Schönfelder. *Sensoren in Wissenschaft und Technik*. 3. [pdf]. Springer Vieweg, 2023. ISBN: 978-3-658-39490-5. DOI: 10.1007/978-3-658-39491-2978-3-662-62697-9.

[Her+24] E. Hering et al. *Elektrotechnik und Elektronik in Maschinenbau und Mechatronik*. 5th ed. Springer, 2024. ISBN: 978-3-662-67538-0.

[Ind25] Indicatorlight. *So verdrahten Sie Kippschalter*. 2025. URL: https://de.indicatorlight.com/faq/toggle-button-switches/.

[MM17] G. Müller and M. Möser. *Ultraschall in Medizin und Technik*. Springer, 2017.

[Mea] *Handbuch MW RD-50A Series*. [pdf]. 2014.

[NKK25] NKK Switches. *Schaltergrundlagen*. 2025. URL: https://www.nkkswitches.de/learning/fundamentals/series_01.html.

[RS 25] RS Components, ed. *MOSFET Leitfaden*. [pdf]. 2025. URL: https://de.rs-online.com/web/content/discovery-portal/produktratgeber/mosfet-leitfaden.

[Raj19] A. Raj. *Arduino Nano 33 BLE Sense Review - What's New and How to Get Started?* 2019. URL: https://circuitdigest.com/microcontroller-projects/arduino-nano-33-ble-sense-board-review-and-getting-started-guide.

[Rei22] Reichelt Online. *Anleitung – COM-MOSFET Modul*. [pdf]. 2022. URL: https://cdn-reichelt.de/documents/datenblatt/A300/COM-MOSFET_ANLEITUNG_2022-07-20.pdf (visited on 05/06/2025).

[Rei25a] Reichelt Online, ed. *COM - MOS-FET*. [pdf]. 2025. URL: https://www.reichelt.de/de/de/shop/produkt/entwicklerboards_-_mosfet_irf9540n-316199 (visited on 05/04/2025).

[Rei25b] Reichelt Online, ed. *Kippschalter - GOOBAY 10013*. [pdf]. 2025. URL: https://www.reichelt.de/de/de/shop/produkt/miniatur-kippschalter_ein-aus_3_a_125_v-359360 (visited on 05/02/2025).

[Rei25c] Reichelt Online, ed. *LED - CML*. [pdf]. 2025. URL: https://www.reichelt.de/de/de/shop/produkt/led-signalleuchte_blau_-_ultrahell_12_vac_vdc_8_mm_600_mcd-271924 (visited on 05/04/2025).

[Rei25d] Reichelt Online, ed. *Schaltnetzteil - MW HRP-75*. [pdf]. 2025. URL: https://www.reichelt.de/de/de/shop/produkt/schaltnetzteil_geschlossen_50_w_5_12_v_6_a-137098#closemodal (visited on 05/02/2025).

[Sau22] M. Sauter, ed. *Grundkurs Mobile Kommunikationssysteme*. 8th ed. Köln, Deutschland: Springer Vieweg, 2022. ISBN: 978-3-658-36963-7.

[Sch13] H. Schulz. *Die Pumpen: Arbeitsweise, Berechnung, Konstruktion*. Springer-Verlag, 2013.

[Sim] *Joy-IT®Ultrasonic Distance Sensor*. SEN-US01. [pdf]. Simac Electronics GmbH. Aug. 2017.

[Sur21] D. Surek. *Pumpen*. Springer, 2021, pp. 1039–1053.

[TM24] P. A. Tipler and G. Mosca. *Tipler Physik*. 9th ed. Springer Spektrum Berlin, May 2024, pp. 423–427. ISBN: 978-3-662-67936-4.

[TR14] H.-R. Tränkler and L. Reindl, eds. *Sensortechnik: Handbuch für Praxis und Wissenschaft*. 2nd ed. VDI-Buch. Published: 13 January 2015. eBook Packages: Computer Science and Engineering (German Language). Berlin, Heidelberg: Springer Vieweg Berlin, Heidelberg, 2014, pp. XIV, 1596. ISBN: 978-3-642-29942-1. DOI: 10.1007/978-3-642-29942-1.

[TR18] H.-R. Tränkler and L. M. Reindl, eds. *Sensortechnik Handbuch für Praxis und Wissenschaft*. 2nd ed. Berlin, Heidelberg: Springer Vieweg, 2018. ISBN: 978-3-642-29942-1.

[Wil22] W. M. Willems. *Lehrbuch der Bauphysik*. 9th ed. Springer Vieweg Wiesbaden, Nov. 2022, pp. 567–568. ISBN: 978-3-658-34093-3.

[goo23] goobay. *Handbuch - goobay 10013*. [pdf]. 2023.

Index

File
.ino, 24
Arduino, 24
Bibliotheken, 27, 29
blink.ino, 38
libraries, 28
MySketch, 24
MySketch.ino, 24
Nano33BLESenseLED, 27, 29
Inertial Measurement Unit
see IMU, 11, 45

General Purpose Input Output
see GPIO, 11, 56
GPIO, 11, 56, 57

I^2C, 11, 56, 57
IMU, 11, 45
Inter-Integrated Circuit
see I^2C, 11, 56

SCL, 11, 56, 57
SDA, 11, 56, 57
Serial Clock
see SCL, 11, 56
Serial Data
see SDA, 11, 56

UART, 11, 56, 57
Universal Asynchronous Receiver Transmitter
see UART, 11, 56

VCC, 11, 56, 57
Voltage at the Common Collector
see VCC, 11, 56