

Programmierung

–

Level

1.1

1.	Vorwort zu den Programmierübungen	3
2.	Ziel der Übung.....	3
3.	Erklärung des gesamten Programms.....	3
4.	Erklärung der verwendeten Befehle	4
2.1.	setup().....	4
2.2.	loop()	4
2.3.	pinMode(pin, modus)	4
2.4.	LED_BUILTIN.....	4
2.5.	digitalWrite(pin, wert)	4
2.6.	delay(ms)	4
5.	Vollständig kommentierter Code	5
6.	Sketch auf den Arduino Nano hochladen	6
5.1.	Arduino IDE starten	6
5.2.	Arduino Nano anschließen.....	6
5.3.	Richtiges Board auswählen	7
5.4.	Richtigen Prozessor auswählen	7
5.5.	COM-Port auswählen	8
5.6.	Code einfügen	8
5.7.	Programm hochladen	9
5.8.	Ergebnis	10

1. Vorwort zu den Programmierübungen

Die folgenden Programmierübungen sind so aufgebaut, dass sie Schritt für Schritt in die Arbeit mit Mikrocontrollern einführen.

Alle Übungen können mit dem Hardware-Einsteigerpaket Level 1 umgesetzt werden und bauen inhaltlich aufeinander auf.

Zu Beginn werden grundlegende Konzepte vermittelt und einfache Aufgaben umgesetzt. Mit jeder weiteren Übung werden neue Funktionen ergänzt und bestehende Inhalte erweitert.

Ziel ist es, ein solides Verständnis für den Aufbau von Programmen, den Umgang mit der Arduino IDE und das Zusammenspiel von Hard- und Software zu entwickeln.

Die Übungen sind bewusst übersichtlich gehalten und sollen dazu ermutigen, eigene Anpassungen vorzunehmen und mit dem Code zu experimentieren.

Es wird empfohlen, die Programmierübungen in der vorgesehenen Reihenfolge durchzuführen, da spätere Aufgaben auf den vorherigen aufbauen.

So entsteht nach und nach ein nachvollziehbarer Lernpfad, der als Grundlage für weiterführende Projekte dient.

2. Ziel der Übung

In dieser Übung lernst du:

- was ein Arduino Sketch ist
- wie ein Programm aufgebaut ist (setup() und loop())
- wie ein Ausgang geschaltet wird
- wie Zeitverzögerungen funktionieren

Am Ende blinkt die LED auf dem Arduino Nano regelmäßig.

3. Erklärung des gesamten Programms

In dieser Übung:

- wird im setup() der LED-Pin als Ausgang festgelegt
- im loop() wird die LED ein- und ausgeschaltet
- zwischen den Zustandswechseln wird jeweils 0,2 Sekunden gewartet

Dadurch entsteht ein gleichmäßiges Blinken.

4. Erklärung der verwendeten Befehle

2.1.setup()

- wird **einmal** beim Start ausgeführt
- hier werden Einstellungen vorgenommen
- z.B. Pins konfigurieren

2.2.loop()

- wird **ständig wiederholt**
- enthält die eigentliche Programmlogik

2.3.pinMode(pin, modus)

Mit pinMode() wird festgelegt, wie ein Pin verwendet wird.

- OUTPUT → der Pin gibt ein Signal aus
- INPUT → der Pin liest ein Signal ein

In dieser Übung wird der LED-Pin als **Ausgang** verwendet.

2.4.LED_BUILTIN

LED_BUILTIN ist ein vordefinierter Name für die **eingebaute LED** auf dem Arduino-Board.

Beim Arduino Nano ist das normalerweise **Pin 13**.

Der Vorteil:

- der Code ist besser lesbar
- er funktioniert auch auf anderen Arduino-Boards

2.5.digitalWrite(pin, wert)

Mit digitalWrite() wird ein digitaler Pin geschaltet.

- HIGH → Pin an (Spannung liegt an)
- LOW → Pin aus (keine Spannung)

2.6.delay(ms)

delay() pausiert das Programm für eine bestimmte Zeit.

- Die Zeit wird in **Millisekunden** angegeben
- 200 bedeutet **0,2 Sekunden**

Während dieser Zeit macht der Mikrocontroller nichts anderes.

5. Vollständig kommentierter Code

```
// Die setup()-Funktion wird einmal beim Start des Programms ausgeführt
void setup() {

    // Der Pin der eingebauten LED wird als Ausgang festgelegt
    pinMode(LED_BUILTIN, OUTPUT);
}

// Die loop()-Funktion läuft nach dem Start ständig in einer Endlosschleife
void loop() {

    // Schaltet die LED ein (Pin bekommt Spannung)
    digitalWrite(LED_BUILTIN, HIGH);

    // Wartet 200 Millisekunden (0,2 Sekunden)
    delay(200);

    // Schaltet die LED aus (keine Spannung am Pin)
    digitalWrite(LED_BUILTIN, LOW);

    // Wartet erneut 200 Millisekunden
    delay(200);
}
```

6. Sketch auf den Arduino Nano hochladen

5.1. Arduino IDE starten

Starte die Arduino IDE durch das Klicken auf das Icon „Arduino IDE“

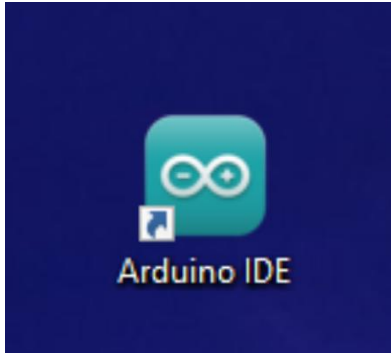


Abbildung 1

5.2. Arduino Nano anschließen

- Verbinde den Arduino Nano mit einem USB-Kabel
- Warte, bis das Betriebssystem das Gerät erkennt

Angeschlossene Mikrocontroller sind im Geräte-Manager unter Anschlüsse (COM&LPT) zu finden.

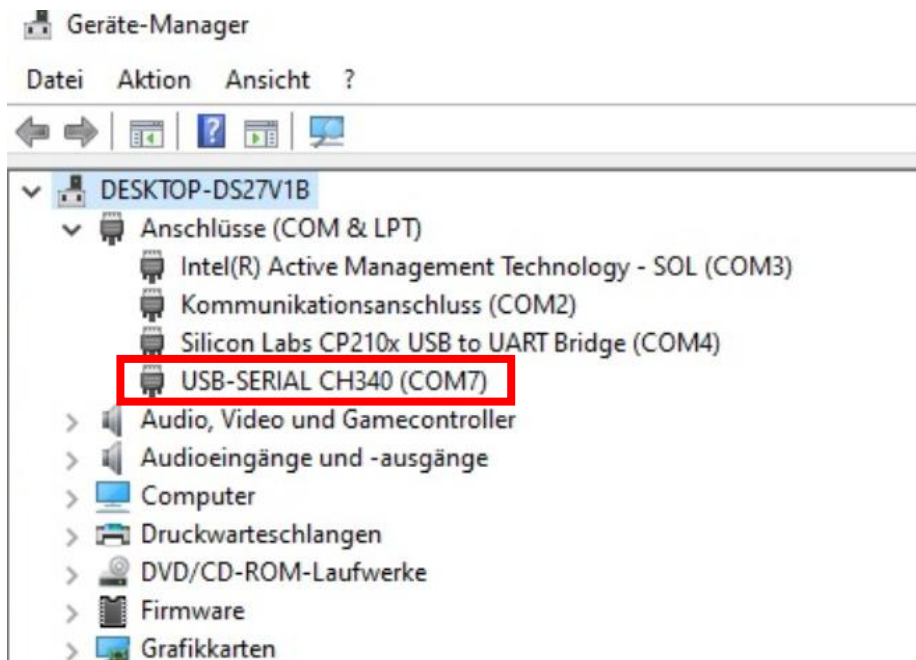


Abbildung 2

5.3. Richtiges Board auswählen

In der Arduino IDE:

- **Werkzeuge** → **Board** → **Arduino AVR Boards** → **Arduino Nano**

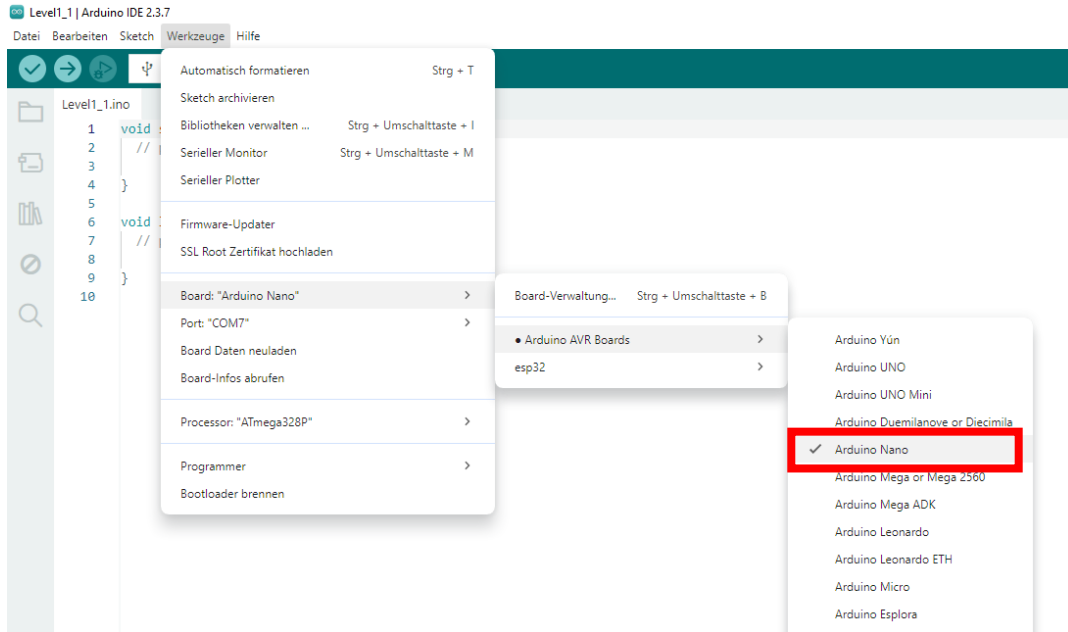


Abbildung 3

5.4. Richtigen Prozessor auswählen

Je nach Nano-Version:

- **Werkzeuge** → **Prozessor**
 - meist: ATmega328P
 - bei älteren Boards evtl.: ATmega328P (Old Bootloader)

Wenn der Upload später fehlschlägt, hier die andere Option testen.

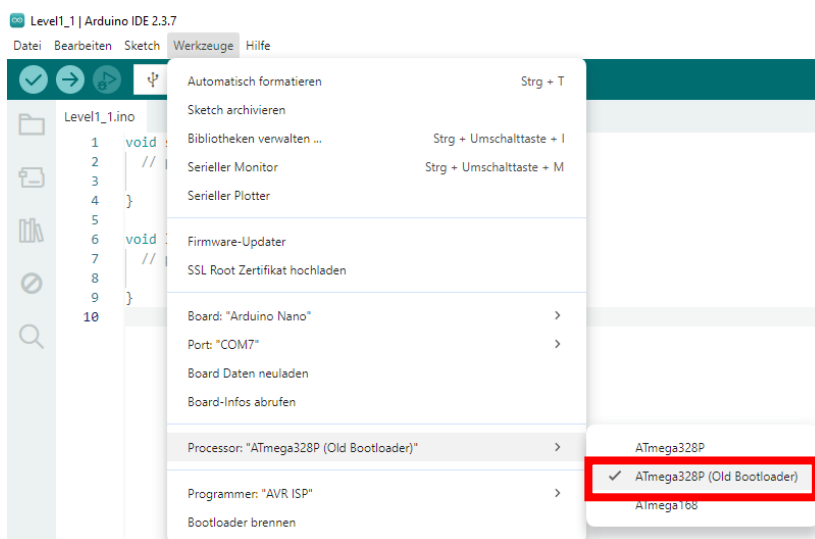


Abbildung 4

5.5.COM-Port auswählen

- **Werkzeuge** → **Port**
- Wähle den Port, an dem dein Arduino Nano angeschlossen ist.
Du findest die Angabe zum COM-Port in deinem Geräte-Manager.

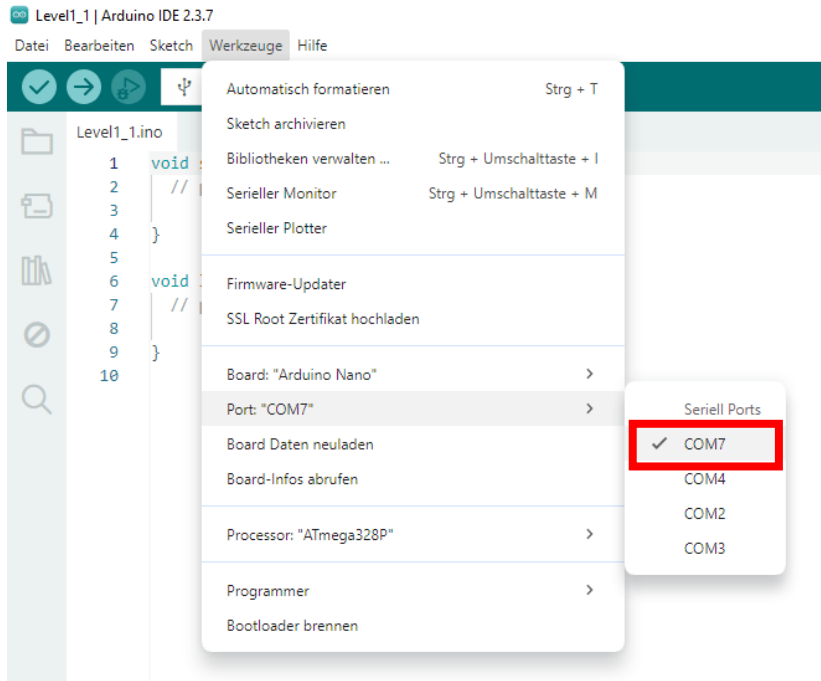


Abbildung 5

5.6.Code einfügen

- Kopiere den Sketch in ein neues Arduino-Projekt
- Speichere das Projekt (z.B. Level1_1_LED_Blinken)

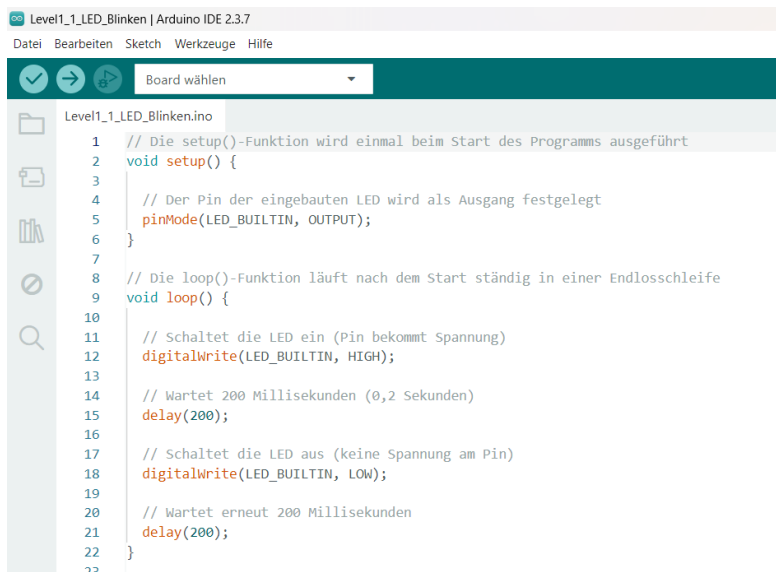


Abbildung 6

5.7. Programm hochladen

- Klicke auf **Hochladen** (Pfeil-Symbol)

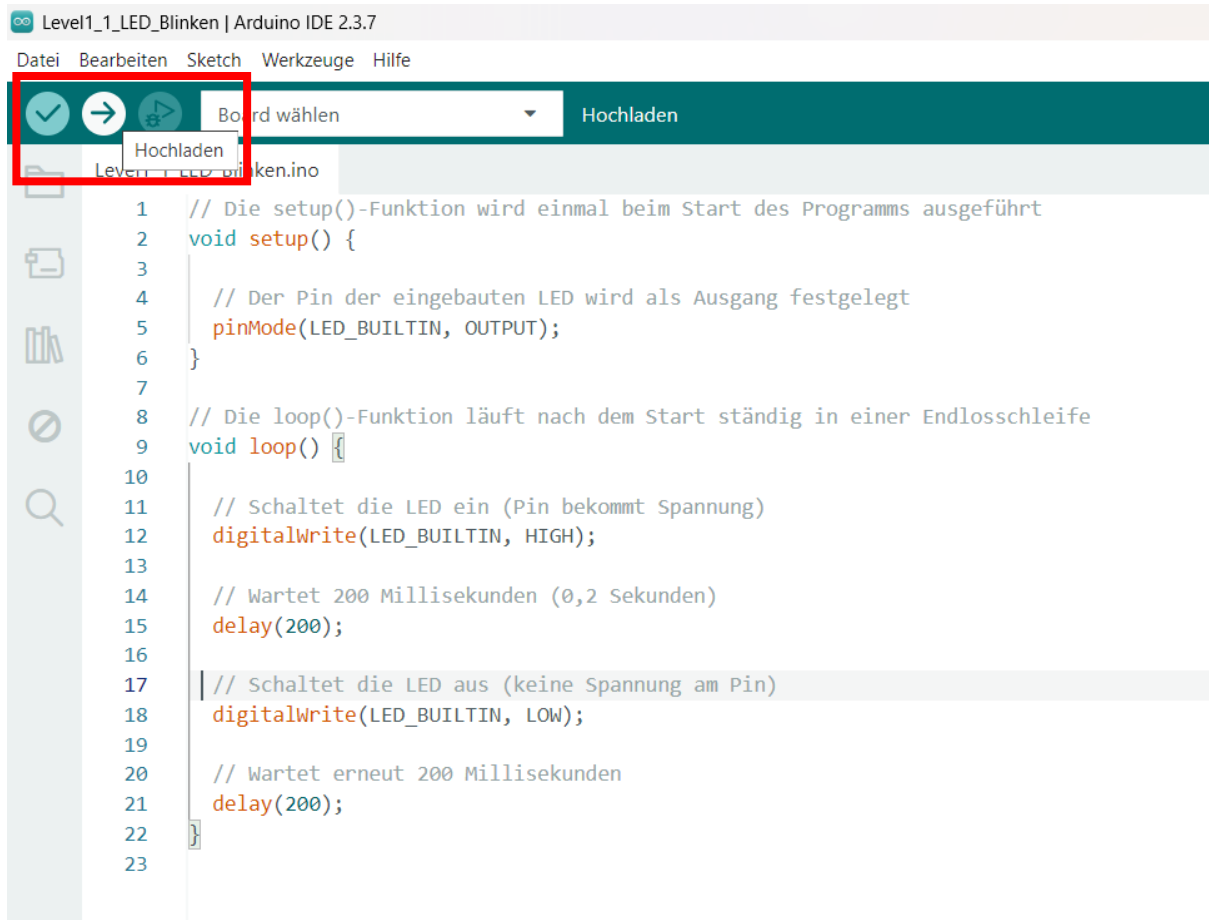


Abbildung 7

- Warte, bis „Upload abgeschlossen“ angezeigt wird

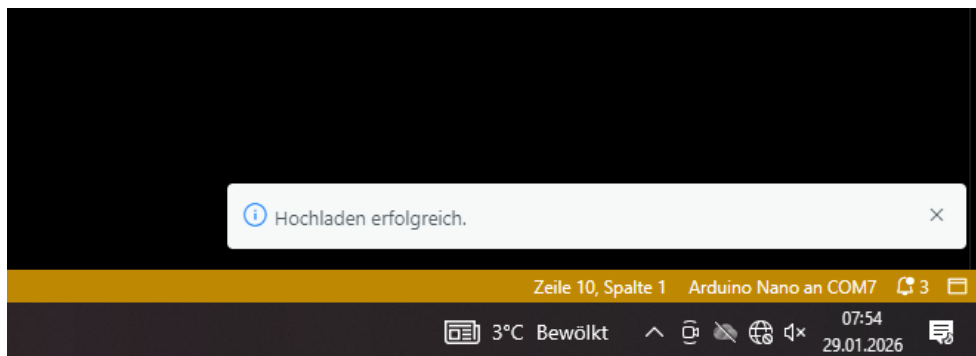


Abbildung 8

5.8.Ergebnis

Nach dem Hochladen:

- beginnt die LED auf dem Arduino Nano zu blinken
- das Programm läuft selbstständig