

Programmierung

–

Level

1.2

1. Vorwort zu den Programmierübungen	3
2. Ziel der Übung.....	3
3. Was ist auf der ATC-Platine bereits vorbereitet?.....	3
4. Neue Befehle/Konzepte in dieser Übung	4
5. Schritt-für-Schritt: Schreibe den Code selbst	4
5.1. Schritt 1 – Pins festlegen (du schreibst zuerst nur das)	4
5.2. Schritt 2 – setup(): Ausgänge konfigurieren	4
5.3. Schritt 3 – Variante A: LEDs blinken abwechselnd	4
5.4. Schritt 4 – Variante B: LEDs blinken gleichzeitig	5
5.5. Schritt 5 – Bonus: Zwei LEDs unterschiedlich schnell.....	5
6. Typische Fehler & schnelle Diagnose	6
7. Referenzlösung Variante A (nur wenn du festhängst)	7
8. Referenzlösung Variante B (nur wenn du festhängst)	8

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 Programmierübung lernst du, zwei externe LEDs auf der ATC-Level-1-Platine anzusteuern.

Du erweiterst deinen Code aus Level 1.1 so, dass mehrere Ausgänge kontrolliert geschaltet werden können.

Am Ende kannst du:

- zwei LEDs abwechselnd blinken lassen
- zwei LEDs gleichzeitig blinken lassen
- (Bonus) zwei LEDs unterschiedlich schnell blinken lassen – und verstehst, warum `delay()` dabei an Grenzen kommt

3. Was ist auf der ATC-Platine bereits vorbereitet?

Auf der ATC-Level-1-Platine sind zwei LEDs so vorgesehen, dass du sie direkt per Code ansteuern kannst:

- **LED 1** ist an **Pin D5**
- **LED 2** ist an **Pin D6**

Du musst in dieser Übung nichts verdrahten.

Wenn deine Platine bestückt ist und der Arduino Nano korrekt steckt, kannst du direkt programmieren.

4. Neue Befehle/Konzepte in dieser Übung

Du verwendest die selben Grundlagen wie in 1.1 und ergänzt nur:

- mehrere Pins als Variablen/Konstanten verwalten
- mehrere `digitalWrite()` im selben Ablauf

Hinweis:

Wir arbeiten hier bewusst noch mit `delay()`.

Später lösen wir „mehrere Dinge gleichzeitig“ sauber mit `millis()`.

5. Schritt-für-Schritt: Schreibe den Code selbst

5.1. Schritt 1 – Pins festlegen (du schreibst zuerst nur das)

Erstelle ein neues Sketch und schreibe oben diese zwei Zeilen:

- eine Konstante/Variable für LED 1
- eine Konstante/Variable für LED 2

Aufgabe:

Nutze die Pin-Nummern 5 und 6.

Selbstcheck: Du hast jetzt zwei Namen im Code, z.B. `LED1_PIN` und `LED2_PIN`.

5.2. Schritt 2 – `setup()`: Ausgänge konfigurieren

In `setup()` musst du beide Pins als **OUTPUT** setzen.

Aufgabe:

Schreibe zwei `pinMode(...)` Zeilen.

Selbstcheck: Du hast in `setup()` zwei `pinMode`-Aufrufe – einen für LED1, einen für LED2.

5.3. Schritt 3 – Variante A: LEDs blinken abwechselnd

Jetzt programmierst du in `loop()` folgendes Muster:

1. LED 1 an, LED 2 aus → kurze Pause
2. LED 1 aus, LED 2 an → kurze Pause

Aufgabe:

Schreibe genau diese Reihenfolge mit `digitalWrite()` und `delay()`.

Tipp (ohne fertigen Code):

- pro Zustand brauchst du 2× `digitalWrite`
- und danach 1× `delay`

Selbstcheck:

Die LEDs wechseln sich sichtbar ab: links/rechts/links/rechts...

5.4. Schritt 4 – Variante B: LEDs blinken gleichzeitig

Jetzt änderst du `loop()` so, dass beide LEDs gleichzeitig schalten:

1. beide an → Pause
2. beide aus → Pause

Aufgabe:

Passe deine `digitalWrite()` Zeilen entsprechend an.

Selbstcheck: Beide LEDs gehen synchron an/aus.

5.5. Schritt 5 – Bonus: Zwei LEDs unterschiedlich schnell

Jetzt kommt eine Aufgabe, die zeigt, warum blockierende Verzögerungen problematisch werden.

Ziel:

- LED 1 blinkt schnell
- LED 2 blinkt langsam

Wichtig:

Mit `delay()` geht das nur „irgendwie“ – es wird schnell unübersichtlich.

Du darfst diese Bonusaufgabe als Challenge behandeln.

Aufgabe (Hinweis statt Lösung):

- Du brauchst zwei verschiedene Zeitwerte (z.B. „kurz“ und „lang“)
- Du musst überlegen, in welcher Reihenfolge du schaltest, damit beide Muster sichtbar sind

Selbstcheck:

Du bekommst es hin, aber du merkst: „Das ist nicht elegant.“

Genau dafür nutzen wir später `millis()`.

6. Typische Fehler & schnelle Diagnose

- **LEDs reagieren gar nicht:**
Prüfe, ob du in `setup()` beide Pins als OUTPUT gesetzt hast.
- **Nur eine LED reagiert:**
Prüfe Pin-Nummern (D5/D6) und ob du beide Pins im Code verwendest.
- **LEDs verhalten sich „komisch“:**
Prüfe die Reihenfolge der `digitalWrite()` und ob du irgendwo vergessen hast, einen Zustand zu setzen.

7. Referenzlösung Variante A (nur wenn du festhängst)

Hinweis: Schau erst hier rein, wenn du es wirklich nicht gelöst bekommst.

// ATC Level 1 – Programmierübung 1.2 (Referenzlösung)

// Pins der ATC-Platine

const int LED1_PIN = 5; // D5

const int LED2_PIN = 6; // D6

```
void setup() {  
  pinMode(LED1_PIN, OUTPUT);  
  pinMode(LED2_PIN, OUTPUT);  
}
```

```
void loop() {  
  // Variante A: abwechselnd blinken  
  digitalWrite(LED1_PIN, HIGH);  
  digitalWrite(LED2_PIN, LOW);  
  delay(500);  
  
  digitalWrite(LED1_PIN, LOW);  
  digitalWrite(LED2_PIN, HIGH);  
  delay(500);  
}
```

8. Referenzlösung Variante B (nur wenn du festhängst)

Hinweis: Schau erst hier rein, wenn du es wirklich nicht gelöst bekommst.

// ATC Level 1 – Programmierübung 1.2 (Referenzlösung)

// Pins der ATC-Platine

const int LED1_PIN = 5; // D5

const int LED2_PIN = 6; // D6

```
void setup() {  
  pinMode(LED1_PIN, OUTPUT);  
  pinMode(LED2_PIN, OUTPUT);  
}
```

```
void loop() {  
  // Variante B: gleichzeitig blinken  
  digitalWrite(LED1_PIN, HIGH);  
  digitalWrite(LED2_PIN, HIGH);  
  delay(500);
```

```
  digitalWrite(LED1_PIN, LOW);  
  digitalWrite(LED2_PIN, LOW);  
  delay(500);  
}
```