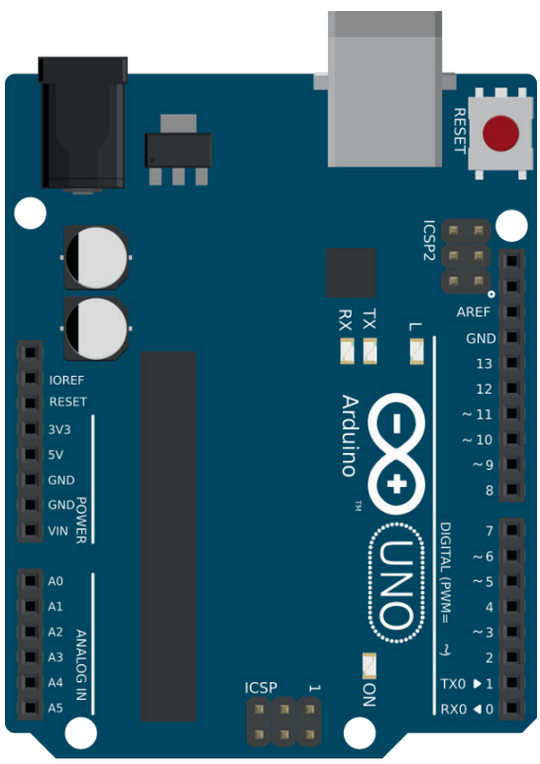
letsgoING

Mechatronik Hochschule Reutlingen

1 - Mikrocontroller kennenlernen



EVA-Prinzip • Mikrocontroller und -board • Arduino IDE • ArduBlock • Algorithmen

gefördert von der vector Stiftung

1.1 Das EVA-Prinzip

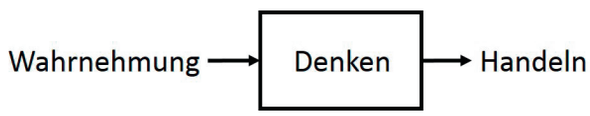
Was ist das EVA-Prinzip

Etwas geht rein (Eingabe oder Input).
Etwas geht raus (Ausgabe oder Output).
Und dazwischen passiert etwas, das wir nicht sehen können (die Verarbeitung).

Diesen Ablauf bezeichnet man als EVA-Prinzip.

Wie funktioniert das beim Mensch?

Man sagt, der Mensch habe fünf Sinne: Sehen, Hören, Schmecken, Riechen und Fühlen. Mit diesen Sinnen bekommst du die Eingaben. Dein Gehirn sorgt beim Denken für die Verarbeitung. Die Ausgabe kann dann z.B. über die Sprache oder Bewegung (Mimik, Klatschen, Lachen, …) erfolgen.



1. **E**ingabe: Du hörst deinen Namen
2. **V**erarbeitung: Du überlegst, wer deinen Namen gesagt hat und ob du antworten willst
3. **A**usgabe: Du fragst nach, was die Person von dir will

Vergleich Mensch und Rechner

Eingabe: Die Sinnesorgane eines Menschen sind Sensoren an einem Mikrocontroller

Verarbeitung: Das Gehirn entspricht der Recheneinheit im Mikrocontroller

Ausgabe: Die Sprache oder Muskeln eines Menschen sind die Aktoren an einem Mikrocontroller

Mensch	Beispiele für Sensoren
Auge	Helligkeitssensor, Farbsensor oder Kamera
Ohren	Lautstärkesensoren, Mikrofone
Haut/Tastsinn	Kraftsensoren, Touch-Sensoren, Taster

Mensch	Beispiele für Aktoren
Muskeln	Motor, Servo, Zylinder
Stimmbänder	Lautsprecher, Buzzer

Erläuterung der Textboxen

Erklärungen

In den grünen Boxen findest du die Erklärungen der Zusammenhänge und Funktionsweisen. Hier steht alles, was du für die Übungsaufgaben und zum Arbeiten mit den einzelnen Inhalten benötigst.

Troubleshooting

In den Rot markierten Boxen findest du immer das Troubleshooting (Fehlerbehebung).
Hier bekommst du Tipps und Hinweise zur selbstständigen Fehlersuche falls etwas mal nicht funktioniert....

Info-Box

Hier werden neue Hardware-Module kurz mit den wichtigsten Infos vorgestellt. Wenn du mehr Informationen zu den Modulen willst, kannst du dir die zugehörigen Projektmodule ansehen.

Übungsaufgaben

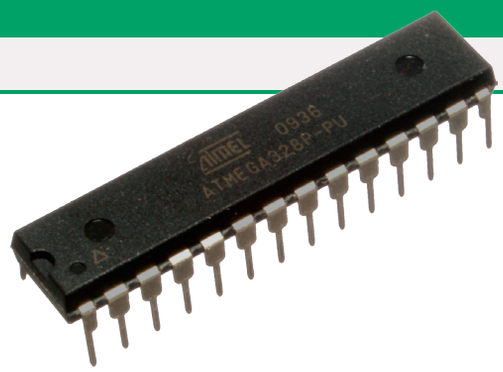
Hier stehen die Übungsaufgaben zu den einzelnen Kapiteln.

Wissensfragen

Hier stehen die Wissensfragen zu den Lernabschnitten.

1.2 Der Mikrocontroller

Was ist ein Mikrocontroller?



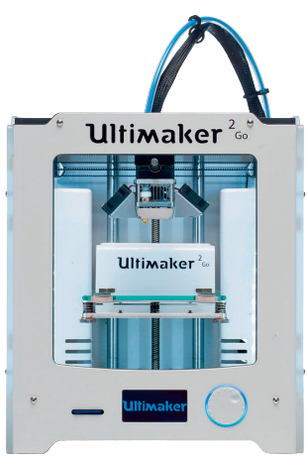
Ein Mikrocontroller ist ein Chip der einen Rechner enthält. Man kann ihn programmieren, also ein Programm auf den Mikrocontroller laden und es darauf laufen lassen. Das Programm kann z.B. eine einfache Rechnung sein, es kann aber auch Geräte und Maschinen steuern. Damit das geht, hat der Mikrocontroller Ein- und Ausgänge (die Beinchen an dem Chip) über die er an sein Umfeld angeschlossen werden kann.

Wo ist der Unterschied zum PC?

Die Grundprinzipien eines Mikrocontrollers und eines PCs sind sehr ähnlich.
Sie unterscheiden sich aber auch in einigen Punkten sehr stark.

Funktion	PC	Mikrocontroller
Eingabe	Tastatur, Maus, Touchpad …	Sensoren für Licht, Temperatur, o.ä. , Lichtschranken, Taster, …
Ausgabe	Monitor, Lautsprecher	LEDs, Digitalanzeigen, Buzzer
Interaktion	Für die direkte Bedienung durch den Mensch ausgelegt	Keine direkte Bedienung - nur wenn im Programm vorgesehen (z.B. Taster soll gedrückt werden).
Software	Viele verschiedene Programme - auch gleichzeitig laufend, Programme können installiert werden	Nur ein Programm zur gleichen Zeit, kann Aufgaben nur nacheinander lösen.
Rechenleistung	durchschnittlich 2-4 x 2,5GHz	1-200MHz (unserer 16MHz)
Stromverbrauch	Ca. 150W (nur PC)	Ca. 0,1W
Preis	Ab 200€	Ab 0,50€

Was kann ein Mikrocontroller besonders gut?



- Einfache Aufgaben sehr schnell lösen
  (Addition zweier Zahlen - ca. 0,2 tausendstel Sekunden)
- Viele Aufgaben hintereinander abarbeiten
- Er ist sehr günstig
- Viel Rechenleistung bei wenig Stromverbrauch
- Immer wiederkehrende Aufgaben bewältigen

→ Immer dann sehr gut, wenn etwas autonom (selbstständig) gemacht werden muss

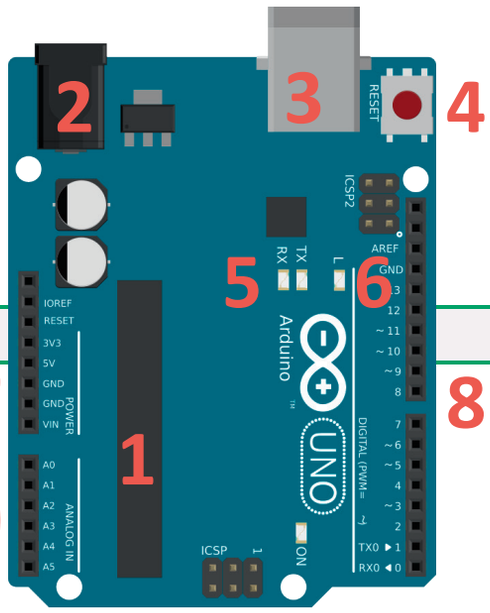
Anwendungsgebiete sind u.a.:
- Haushaltsgeräte wie Waschmaschinen, Spülmaschinen, Kaffeemaschinen,…
- Uhren, Telefone, Messgeräte, …
- Spielzeug wie ferngesteuerte Autos, Drohnen, …
- Roboter, 3D-Drucker, Lasercutter, CNC-Fräsen, …

1.3 Das Mikrocontroller-Board

Warum brauche ich ein Mikrocontoller-Board?

Ein Mikrocontroller-Board hat viele Vorteile gegenüber dem blanken Mikrocontroller. Das Board bietet:

- Eine Schnittstelle zum Programmieren mit dem PC
- Die Stromversorgung über Batterie oder USB
- Ein einfaches Einstecken von Sensoren und Aktoren
- Einen mechanischen Schutz für den Mikrocontroller (z.B. Schutz vor verbogenen oder abgebrochenen Pins)



Was ist auf dem Board?

1 **Mikrocontroller:** Das „Gehirn“ des Mikrocontroller-Boards
2 **DC-Buchse:** Hier kann man Batterien und Netzteile von 7-12V einstecken
3 **USB-Anschluss:** Hier kann der Mikrocontroller programmiert und mit Strom versorgt werden
4 **Reset Taster:** Startet das Programm auf dem Controller neu
5 **Kommunikations-LEDs:** Hier kann man z.B. erkennen ob gerade ein Programm hochgeladen wird
6 **Onboard-LED:** Einfache LED, die mit dem Mikrocontroller angesteuert werden kann
7 **Pin-Leiste Strom:** Hier kann man Sensoren und Aktoren mit Strom versorgen
8 **Pin-Leisten digitale Ein-/Ausgänge:** Verarbeitung von digitalen Signalen → Wird später erläutert
9 **Pin-Leiste analoge Eingänge:** Verarbeitung von analogen Eingangssignalen → Wird später erläutert

Was sind die wichtigsten Daten des Boards?

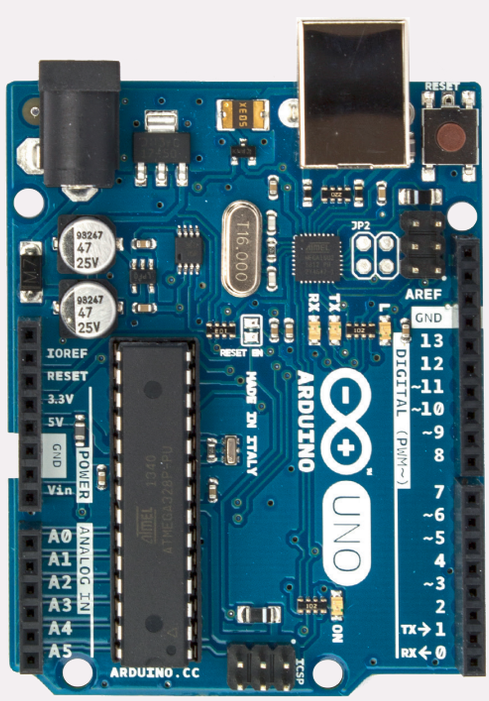
Name:	Arduino/Genuino UNO
Mikrocontroller:	Atmel Atmega328p (16Mhz)
Spannung:	5V
Ein-/Ausgänge digital:	14
PWM-Ausgänge:	6
Analoge Eingänge:	6

Die Arduino-Familie

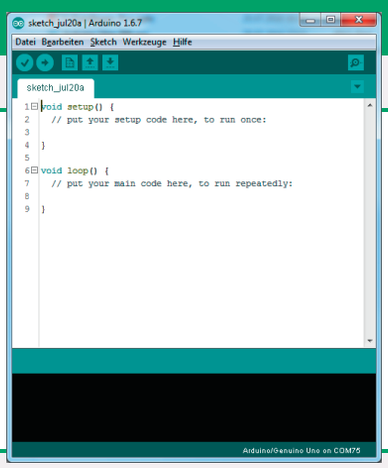
Es gibt neben dem Arduino UNO auch noch sehr viele andere Mikrocontroller-Boards. Jedes dieser Boards hat andere Spezialitäten, wie z.B.:

- sehr viele Ein-/Ausgänge
- einen schnelleren Mikrocontroller
- ist besonders klein gebaut
- oder hat besondere Anschlüss wie z.B. WLAN

Alle diese Boards können auch mit der Arduino IDE (nächstes Kapitel) und ArduBlock (übernächstes Kapitel) programmiert werden.



1.4 Die Arduino IDE



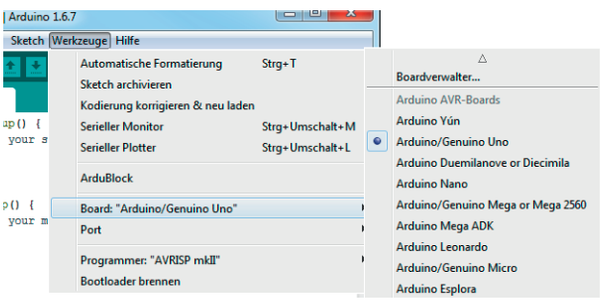
Was ist die Arduino IDE?

Die Arduino IDE ist das Programm mit dem du dein Arduino-Board programmieren kannst.
IDE ist eine englische Abkürzung für Integrierte Entwicklungsumgebung.
Integriert bedeutet, dass du alle Schritte die nötig sind um dein Programm auf den Mikrocontroller zu bekommen, mit nur einem Programm durchführen kannst. Die meisten Schritte macht eine IDE ganz selbstständig.

→ Früher benötigte man 4 Programme, die man nacheinander ausführen musste.

Wie lade ich Programme auf den Arduino?

Das Programm muss im Programmierfenster geschrieben werden (dazu musst du die Programmiersprache „C++“ verwenden). Wenn das Programm fertig ist und auf den Controller geladen werden soll, muss dazu nur der Hochladen-Button gedrückt werden. Wenn alles gut geht, steht in der Info-Leiste nach einiger Zeit „Hochladen abgeschlossen“. Wenn es nicht funktioniert hat, kannst du Hinweise zu den Ursachen im Nachrichtenfenster finden.

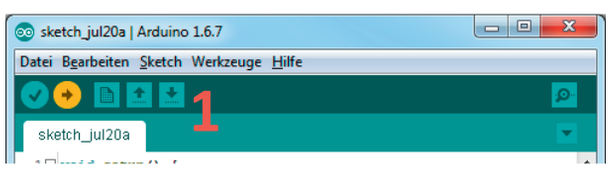


Vor dem ersten Hochladen müssen noch folgende Einstellungen gemacht werden:
„**Werkzeuge → Board**“: Arduino/Genuino UNO
„**Werkzeuge → Port**“: COM-Port des verwendeten Arduinos
(ggf. im Windows- Menü „Geräte und Drucker“ nachsehen)
„**Werkzeuge → Programmer**“: AVRISP mkII

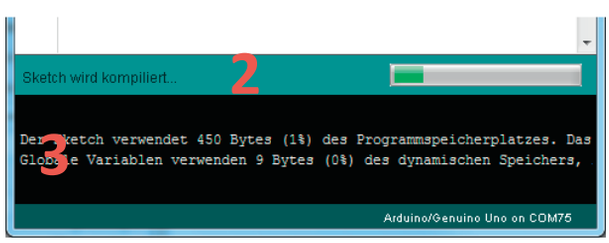
Was finde ich wo?

Die Oberfläche der Arduino IDE hat drei wichtige Bereiche.

1. Die Buttonleiste
- **Überprüfen:** Programm auf Fehler prüfen
- **Hochladen:** Programm auf Fehler prüfen, anschließend auf den Mikrocontroller laden
- **Neu:** neues Programmfenster öffnen
- **Öffnen:** gespeichertes Programm öffnen
- **Speichern:** aktuelles Programm speichern



2. Die Aktivitätenleiste
An der Aktivitätenleiste kannst du erkennen ob die IDE gerade dein Programm übersetzt oder hochlädt und wie weit sie ist.



3. Das Nachrichtenfenster
Ganz unten in der Arduino-IDE befindet sich das Nachrichtenfenster. Hier wird angezeigt, ob das Hochladen erfolgreich war oder ob Fehler aufgetreten sind. Die Fehlermeldungen geben einen Hinweis, wo und was für ein Problem aufgetreten ist.

Übungsaufgabe

Übung

Öffne unter „Datei“→“Beispiele“→“„Basic“ das Blink-Programm und lade es auf den Arduino. Beobachte die Onboard LED.

Falls etwas nicht funktioniert, findest du im Troubleshooting-Block Hinweise auf die häufigsten Fehler und wie du sie beheben kannst.

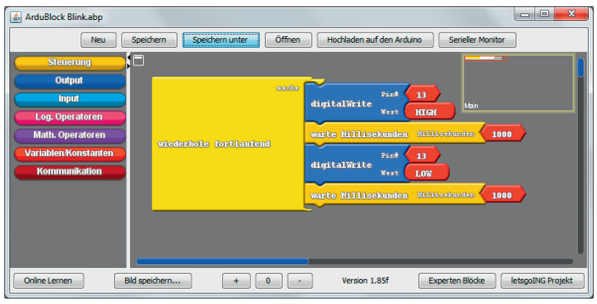
Troubleshooting: Häufige Fehler

ser_open(): can‘t open device:
Arduino-COM-Port falsch eingestellt oder Arduino nicht eingesteckt
stk500_getsync() not in sync:
Arduino ist nicht erreichbar, evtl. USB überlastet
→ aus- und wieder einstecken
exit status 1: Fehler im Programm, weitere Hinweise beachten

1.5 ArduBlock

Was ist ArduBlock?

ArduBlock ist eine grafische Programmieroberfläche. In verschiedenen Menüs stehen dir Blöcke für die Programmierung zur Verfügung. Du brauchst dich also nicht um die Schreibweise der Befehle zu kümmern. Dadurch hast du einen leichteren Einstieg und kannst dich voll auf die Funktion deines Programms
konzentrieren.
ArduBlock ist auf die Arduino IDE aufgesetzt und nutzt diese zum Hochladen und zur Fehlerprüfung. Daher kannst du den „Hochlade-Status“ und Fehlermeldungen nur im Nachrichtenfenster der Arduino-IDE sehen.
ArduBlock wird über das Arduino-Menü „Werkzeuge“ gestartet.

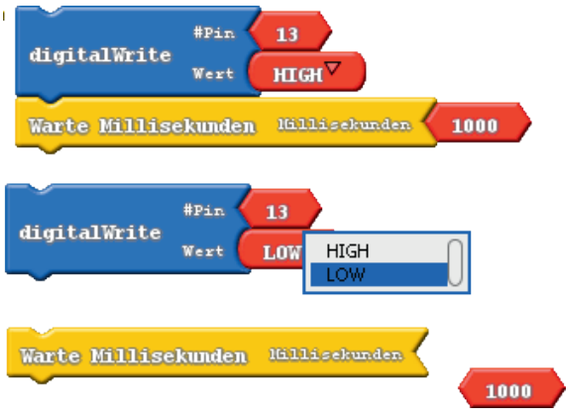


Wie programmiere ich mit ArduBlock?

Mit ArduBlock programmierst du, indem du einzelne **Blöcke aneinander heftest oder** sie **miteinander verbindest**. Dabei kannst du dir die gewünschte Funktion in den seitlichen Menüs heraussuchen. Bei einigen Blöcken muss man noch etwas einstellen oder eine Zahl eingeben.

Wenn mehrere Blöcke miteinander verbunden werden sollen, musst du darauf achten, dass die Formen passen. Jede **Form** steht für eine bestimmte Art von Information bzw. Signal.

Anhand der **Farben** der Blöcke kannst du erkennen, ob ein Block für einen Ausgang, einen Eingang oder andere Funktionen ist. Welche Farbe für was steht, lernst du im Laufe dieses Kurses.



Wie kommt mein Programm auf den Arduino?

Wenn du auf den **„Hochladen“-Button** bei ArduBlock drückst, laufen folgende **5 Schritte** ab:

1. Das ArduBlock-Programm wird in C++ übersetzt und in die Arduino IDE übertragen

2. Das C++-Programm wird auf Fehler überprüft

3. Das überprüfte Programm wird in Maschinenbefehle übersetzt, die der Mikrocontroller versteht

4. Die Maschinenbefehle werden auf den Mikrocontroller hochgeladen

5. Das Programm wird gestartet


```
1  void setup()
2  {
3    pinMode( 13 , OUTPUT);
4  }
5
6  void loop()
7  {
8    digitalWrite( 13 , HIGH );
9    delay( 1000 );
10   digitalWrite( 13 , LOW );
11   delay( 1000 );
12 }
```

Übungsaufgabe

Übung

Baue das Programm rechts in ArduBlock nach und lade es auf den Arduino.

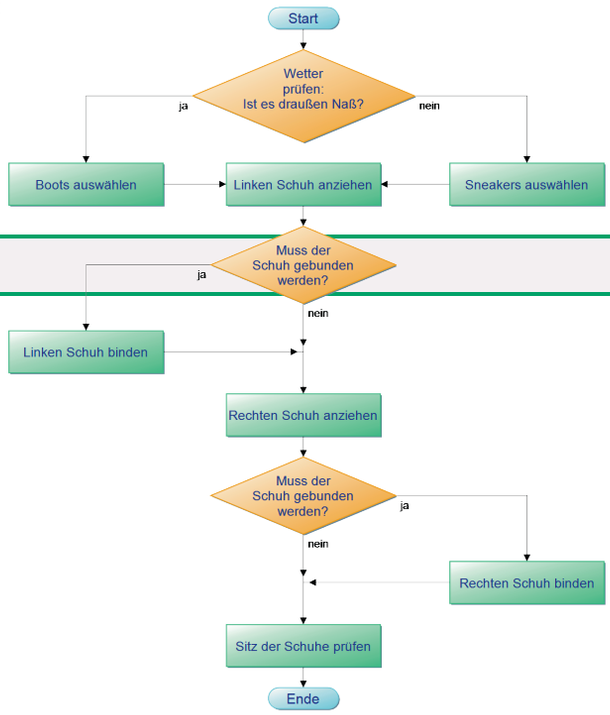
Überprüfe was in der Arduino IDE passiert.
Verändere das Programm so, dass die LED länger an ist als aus.

1.6 Algorithmen

Was ist ein Algorithmus?

Ein Algorithmus ist, ganz allgemein gesagt, eine Schritt für Schritt Anleitung. Algorithmen begegnen dir im Alltag immer wieder in ganz unterschiedlicher Form und werden immer dann benötigt, wenn Dinge in einer bestimmten Reihenfolge und in einer bestimmten Art und Weise passieren sollen.
Sehr gängige Formen sind z.B. Bauanleitungen oder Koch- und Backrezepte. Wenn man beim Backen die Zutaten zuerst in den Ofen stellt, bevor man daraus einen Teig geknetet hat, wird das nicht funktionieren. Ein gutes Beispiel für einen einfachen Algorithmus im Alltag ist auch das Anziehen von Schuhen.

Ein Algorithmus ist also eine Folge von Anweisungen um ein bestimmtes Problem zu lösen.



Algorithmen bei Menschen

Wenn du etwas zum ersten Mal machen willst, brauchst du eine sehr genaue Anleitung. Je nachdem, wie kompliziert die Aufgabe ist, kannst du dir den Ablauf nach ein- oder mehrmaliger Wiederholung merken und dann selbstständig ausführen. Das Verinnerlichen des Ablaufs entspricht im Prinzip der Programmierung des Algorithmus in einen Rechner.

Es gibt drei sehr große Unterschiede zwischen einem Algorithmus in deinem Gehirn und dem auf einem Rechner:
1. Du kannst selbstständig weiter lernen
   z.B. ist dir etwas angebrannt → reduziere nächstes mal die Temperatur
2. Du kannst entscheiden, ob etwas wichtig ist oder nicht (abstrahieren)
   z.B. linken oder rechten Schuh zu erst anziehen → Reihenfolge ist egal
3. Du kannst Erfahrungen aus anderen „Algorithmen“ übertragen
   z.B. schmeckt dir ein Gewürz nicht → verwende es in dem neuen Rezept nicht

Das alles kann ein Rechner mit programmiertem Algorithmus nicht.

Algorithmen bei Maschinen

Bei einerm Rechner muss der Algorithmus einprogrammiert werden. Ist er einmal programmiert, kann der Rechner diesen Algorithmus für (fast) immer wiederholen. Der Rechner kann auch auf äußere Einflüsse reagieren, aber nur wenn es programmiert wurde. Anders als Menschen kann er nichts neues dazulernen.

Algorithmen gibt es in ganz verschiedenen Komplexitätsstufen. Du hast in Aufgabe 1.5 schon einen einfachen Algorithmus programmiert. Auch z.B. die Wettervorhersage wird von einem Algorithmus berechnet. Anders als dein Algorithmus benötigt der Wetteralgorithmus aber extrem viele Informationen und führt sehr viele Berechnungen und Entscheidungen aus um auf ein gutes Ergebnis zu kommen.
Im Alltag begegnen dir Algorithmen vor Allem im Internet und in Apps auf deinem Smartphone. Wenn du z.B. in einem Onlineshop etwas bestellst, werden dir sofort Produkte vorgeschlagen, die andere zusammen mit dem von dir ausgesuchten Produkt gekauft haben.

Lernende Maschinen

Seit einiger Zeit gibt es auch selbstlernende Algorithmen bzw. Maschinen. Dabei wird mit Hilfe von komplizierten Rechenoperationen ein Verhalten programmiert, welches ungefähr dem menschlichen Lernen entspricht. Das nennt sich dann „Künstliche Intelligenz“.

Besondere Beispiele sind der Google-Suchalgorithmus, die Werbevorschläge auf Facebook und auch die Produktvorschläge bei Amazon. Alle drei beobachten und merken sich was du angesehen, geliked und gekauft hast und machen dir dann Vorschläge, die zu deinen Interessen und deinem Verhalten passen.
Das ist erstmal sehr komfortabel, kann aber auch weitreichende Folgen haben.

1.7 Übungsprojekt

Projektbeschreibung

Eine der einfachsten Arten über lange Strecken zu kommunizieren ist das Morsen. Dabei werden Buchstaben durch bestimmte Impulsfolgen dargestellt. Früher wurden so über Telegrafen-Leitungen zum ersten mal über lange Strecken Informationen ausgetauscht.

Heutzutage werden Morsezeichen nur noch in sehr seltenen Fällen eingesetzt. Wenn man z.B. weit ab der Handy-Netze in Gefahr kommt, kann man über das internationale Notrufzeichen „SOS“ Hilfe rufen. Das besondere an Morsezeichen ist, dass man sie sowohl elektrisch, optisch als auch akustisch übertragen kann.

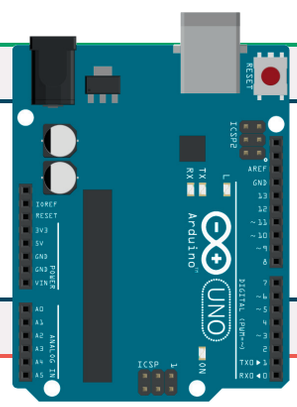
Programmiere nun einen einfachen Algorithmus, der das internationale Notrufzeichen SOS optisch darstellt und die ganze Zeit wiederholt. Du kannst das Signal über die Onboard-LED ausgeben.

Aufgabenstellung

Schreibe ein Programm, welches die Zeichenfolge „SOS“ auf der Onboard-LED ausgeben kann.

- Das Zeichen „S“ besteht aus drei kurzen Blink-Impulsen (z.B. 300 ms)
- Das Zeichen „O“ besteht aus drei langen Blink-Impulsen (z.B. 900 ms)
- Die Pausen sind immer gleich lang wie die kurzen Impulse (z.B. 300ms)
- Nach jedem Wort kommt eine längere Pause (z.B. 1200ms)

Du kannst als Hilfestellung das Blink-Programm aus der Übungsaufgabe 1.5 zur Vorlage nehmen.



Versuchsaufbau

Du brauchst für dieses Übungsprojekt nur deinen Arduino.

Als LED nutzt du, wie in den Blink-Programmen vorher, die Onboard-LED an Pin13.

Troubleshooting

LED geht nicht an / LED geht nicht aus

→ hast du im Programm Pin 13 ausgewählt?
→ wird die LED an und ausgeschaltet (HIGH / LOW)?
→ sind die Pausen an der richtigen Stelle? (immer nach dem „digitalWrite“ Befehl)
→ sind die Pausen lang genug? (der Mensch erkennt mit dem Auge nur Änderungen die langsamer als 20ms sind)

Programm wird nicht hochgeladen

→ sind alle Einstellungen in der IDE richtig? (siehe Kapitel 1.3 Arduino IDE)
→ ist der Arduino eingesteckt?

Zeiten ändern sich nicht

→ hast du das Programm nach der Änderung neu hochgeladen?

Wissensfragen

1. Erkläre anhand eines eigenen Beispiels das EVA-Prinzip beim Menschen.
2. Nenne drei Anwendungsgebiete für einen Mikrocontroller und erkläre weshalb du dabei einen Mikrocontroller einsetzen würdest.
3. Wo lässt sich erkennen, ob der Programmupload erfolgreich war?
4. Erkläre kurz, was ein Algorithmus ist bzw. was ihn ausmacht.