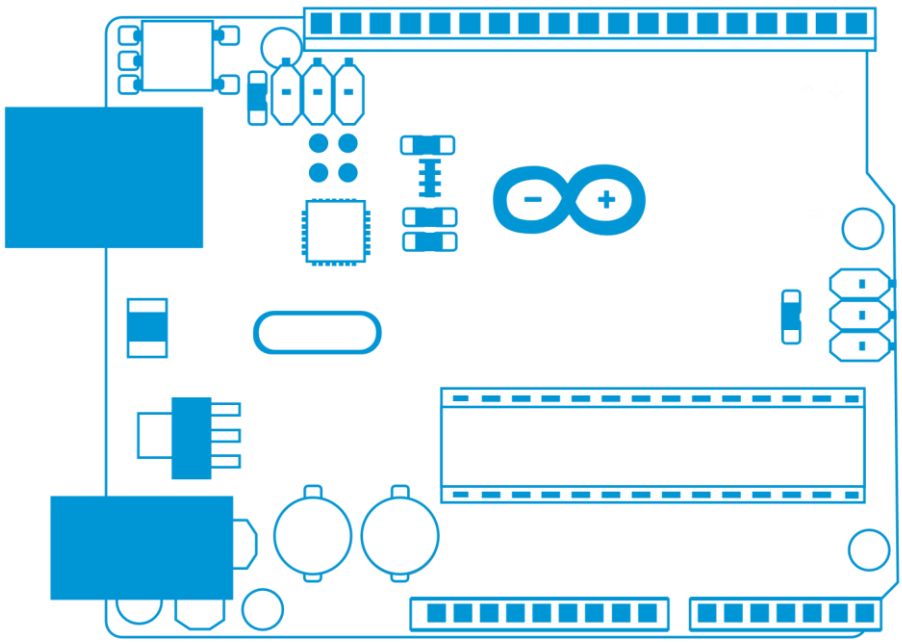
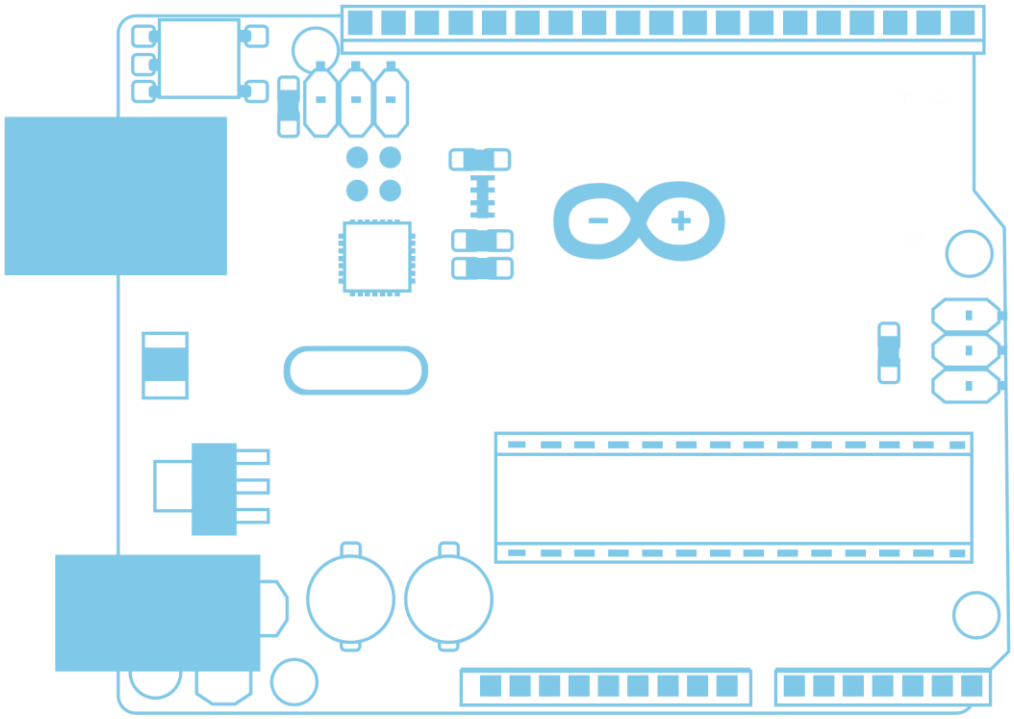
Lehren mit dem All

→ Lernen mit ARDUINO!

Einführung in die Datenverarbeitung mit Arduino in C++





Lehren mit dem All – Lernen mit Arduino | T04.1
www.esa.int/education

Das ESA Education Office freut sich über Feedback und Kommentare
teachers@esa.int

Eine Produktion von ESA Education

Eine Übersetzung von ESERO Germany

→ Lernen mit ARDUINO!

Einführung in die Datenverarbeitung mit Arduino in C++

Kurzinformationen

Alter: 14-20 Jahre

Lehrplanbezug: Programmieren, Elektronik

Schwierigkeitsgrad: Mittel

Benötigte Zeit: 90-120 Minuten

Örtlichkeit: in einem Gebäude

Voraussetzungen: Arduino Software und Hardware

Schlüsselbegriffe: Arduino, Sensor, Quelltext

Überblick

Durch den Umgang mit dem Arduino-Tool lernen die SuS Technologie kennen, die im Weltraum verwendet wird. Sie werden Schaltkreise bauen, die dazu verwendet werden, eine LED zum Leuchten zu bringen und Temperatur, Druck und Höhe zu messen. Der Umgang mit der Arduino DIE-Software (Integrated Development Environment) dient der Einführung in die Grundlagen des Programmierens mit C++. Diese Aktivitäten können ein Anfangspunkt sein für zukünftige Teilnahmen an dem CanSat-Wettbewerb.

Lernziele

- die SuS verbessern ihre analytischen Fähigkeiten.
- die SuS verstehen, wie eine Programmiersprache funktioniert.
- die SuS verstehen, wie man Schaltkreise baut.
- die SuS verstehen, wie Sensoren dazu genutzt werden, um Daten zu sammeln.
- die SuS verbessern ihre Teamfähigkeit.

Zusammenfassung der Aufgaben					
	Bezeichnung	**Beschreibung**	**Ergebnis**	**Voraussetzungen**	**Zeit**
1	Erste Schritte	Eine Einführung in die Komponenten für die zu lösende Aufgabe	Die SuS machen sich vertraut mit den nötigen Komponenten und ihren Funktionen	Grundlegendes Verständnis von elektronischen Komponenten	10 Minuten
1	Mach mich an!	Die SuS bauen ihren ersten Arduino Schaltkreis	Die SuS sind in der Lage, über einen selbst geschriebe-nen Quelltext eine LED zu steuern	Vollendung vorheriger Aufgaben	15 Minuten
2	Signalisiere ein SOS	Die SuS lernen, wie man mithilfe einer LED komplexere Nachrichten verschicken kann	Die SuS sind in der Lage, den Quelltext und das Programm zu schreiben, um mit einer LED ein SOS zu signalisieren	Vollendung vorheriger Aufgaben	15 Minuten
3	Temperatur-messung	Die SuS verwenden einen Thermistor, um die Umge-bungstemperatur zu messen	Die SuS sind in der Lage komplexere Schaltkreise zu bauen und diese dazu zu verwen-den, Messungen ihrer Umwelt vorzunehmen	Vollendung vorheriger Aufgaben	20 Minuten
4	Druckmessung	Den SuS wird gezeigt, wie man einen Drucksensor verwendet, um den gegenwärtigen Luftdruck zu messen	Die SuS sind in der Lage, komplexere Schaltkreise zu bauen und diese dazu zu verwen-den, Messungen ihrer Umwelt vorzunehmen	Vollendung vorheriger Aufgaben	20 Minuten
5	Höhenmessung	Die SuS lernen, die Höhe aus Messun-gen des Umgebungs-luftdrucks abzuleiten	Die SuS sind in der Lage, Schaltkreise zu bauen und Quelltext zu bearbeiten, um Messungen in ihrer Umwelt vorzunehmen	Vollendung vorheriger Aufgaben	20 Minuten

→ Aufgabe 0: Erste Schritte - Einführung

Arduino ist eine Open-Source-Plattform, die dazu verwendet wird, eine große Bandbreite an elektronischen Projekten durchzuführen. Dies beinhaltet sogar die Möglichkeit, echte Weltraummissionen zu simulieren, wie z.B. ExoMars (http://exploration.esa.int/mars/).
Arduino beinhaltet beides, eine physische, programmierbare Leiterplatine (Hardware), sowie eine Software IDE (Integrated Development Environment), die auf einem Computer verwendet werden kann.

Die Arduino Hardware und Software wurden entwickelt für Künstler, Designer, Hobbybastler, Hacker und Menschen, die daran interessiert sind, interaktive Objekte und Umgebungen zu gestalten. Arduino kann dazu verwendet werden mit Knöpfen, LEDs, Sensoren, Motoren, Lautsprechern, GPS-Geräten, Kameras, dem Internet und sogar Smartphones sowie Fernsehern zu interagieren. Diese Vielseitigkeit, gepaart mit den Tatsachen, dass die Arduino Software frei zugänglich ist, dass die Hardware verhältnismäßig günstig ist und dass sowohl Software als auch Hardware intuitiv einfach zu bedienen sind, haben zu einer großen Benutzer-Community geführt (www.arduino.cc).

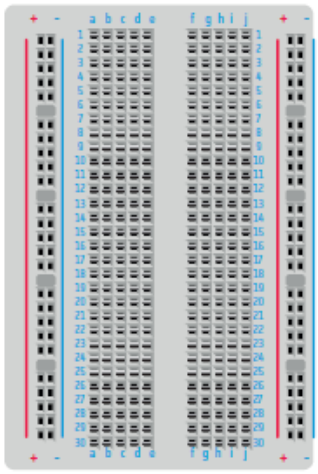
Hardware

Für diese Aufgabe wird das Arduino Uno (Abbildung A1, Kästchen h) verwendet. Das Arduino versorgt seine Pins mit Spannung und misst diese. Beachten Sie die Zeichen, die neben jedem Metallstift abgebildet sind. **GND** (ground) fungiert hier als negativer Pol, **Vin** oder **5V** als positiver (so wie die positive und negative Seite einer Batterie).

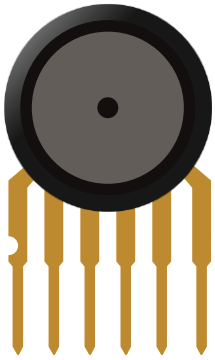
Die folgenden Komponenten sollen verwendet werden (s. Abbildung A1):

a. **Lochrasterplatte:** Eine Trägerplatte auf der sich elektr. Komponenten einfach verbinden lassen.
b. **Drucksensor MPX4115A:** Misst den absoluten Luftdruck der Umgebung.
c. **Widerstand mit 220Ω und 10kΩ:** Zur Kontrolle des Stromflusses. Widerstand wird gemessen in Ohm (Ω). Widerstände sind farblich markiert, um ihren jeweiligen Wert anzuzeigen.
d. **USB-Kabel:** Wird verwendet, um den Arduino an einen Computer oder eine Stromquelle anzuschließen.
e. **Kabel:** Werden verwendet, um Komponenten elektrisch miteinander zu verbinden.
f. **LEDs (Light Emitting Diode):** Elektrische Komponente, die Licht abstrahlt, wenn Strom durch sie hindurchfließt.
g. **Temperatursensoren:** Ein analoger Sensor (wie z.B. der LM35) und ein Thermistor (ein Widerstand, der besonders empfindlich auf Temperatur reagiert).
h. **Arduino Platine:** Funktioniert wie ein einfacher Computer. Verfügt über einen Prozessor, eine Speichereinheit sowie Eingangs- und Ausgangspins.

Abbildung A1



a. Lochrasterplatte

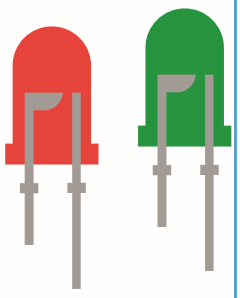


b. Drucksensor

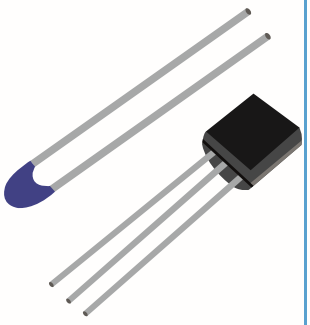
c. Widerstand

d. USB-Kabel

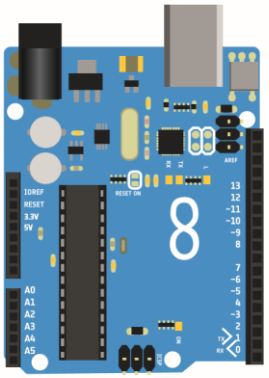
e. Kabel



f. LEDs (Light Emitting Diode)



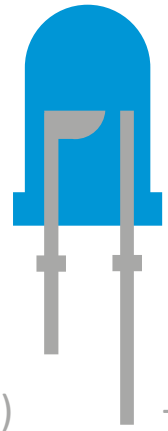
g. Temperatursensoren



i. Arduino Platine

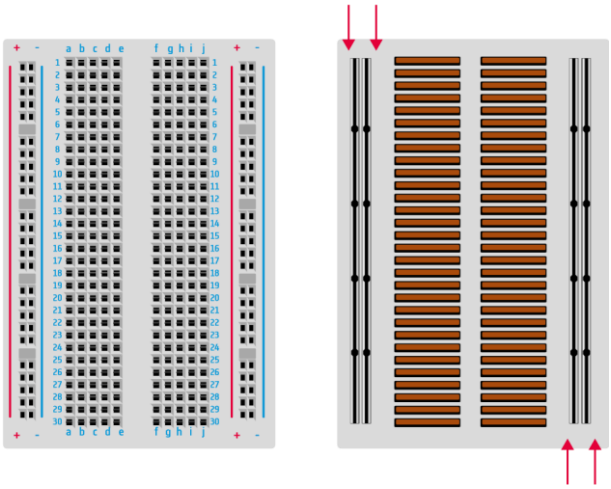
↑ Arduino Grundausstattung

Abbildung A2



↑ LED: Beachten Sie, dass die LED einen langen und einen kurzen Metallstift hat. Der lange ist die Anode, der kurze die Kathode.

Abbildung A3



↑ Beachten Sie, dass alle Metallstifte jeder Reihe miteinander verbunden sind.

Sobald Sie Ihren Arduino mit einem Computer verbinden, sollte der Computer ihn erkennen und die Installation der Treiber veranlassen. Sollten Sie bei der Installation der Treiber auf Probleme stoßen, finden sie hier weitere Informationen: www.arduino.cc/en/Guide/ArduinoUno

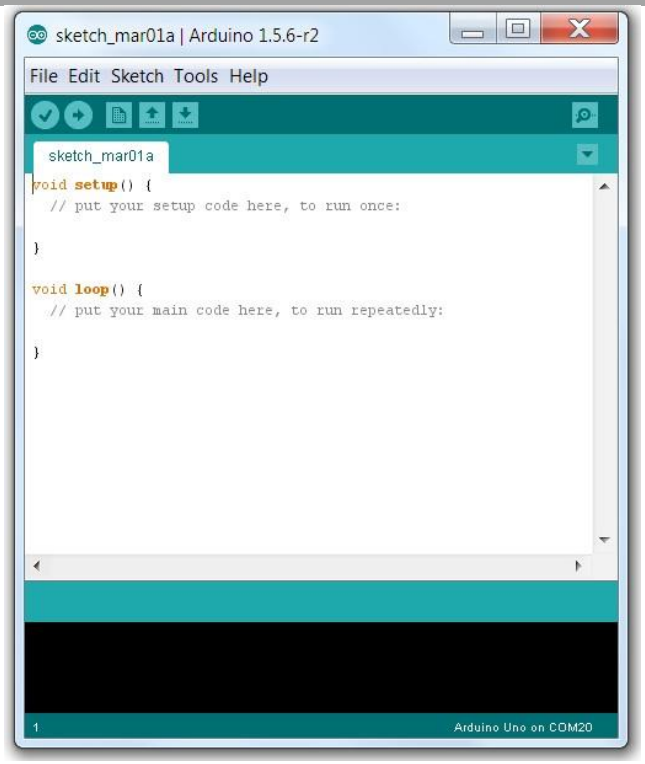
*Stellen Sie sicher, dass sie im Tools-Menü die richtige Platine (Arduino Uno) und die korrekte serielle Anschlussstelle (COM) ausgewählt haben!

Software

Die Arduino-Software kann hier einfach heruntergeladen und installiert werden: www.arduino.cc/en/Main/Software

Programme, die mit der Arduino-Software (IDE) geschrieben wurden, werden Sketches genannt. Diese Sketches werden im Text Editor geschrieben und mit der Dateiendung **.ino** gespeichert. Der Editor verfügt über Werkzeuge zum Ausschneiden / Einfügen und Suchen / Ersetzen von Text. Die Nachrichtenanzeige unten (Abbildung A4) zeigt beim Speichern und Exportieren sowie bei auftretenden Fehlern Rückmeldungen an. Die ausgewählte Platine und die ausgewählte serielle Anschlussstelle werden in der unteren rechten Ecke des Fensters angezeigt. Die Knöpfe der Werkzeugleiste erlauben es Ihnen, Programme zu verifizieren und hochzuladen, Sketches zu erstellen, zu öffnen und zu speichern und den Monitor zu öffnen (um die von den Sensoren gesammelten Daten anzuschauen). Abbildung A4 ist ein Screenshot des graphischen Interfaces der IDE Arduino.

Abbildung A4



↑ Arduino IDE interface

Arduino Befehle - Tabelle A1

	**Verify:** Überprüft Ihren Quelltext auf Fehler, während er zusammengestellt wird.
	**Upload:** Stellt Ihren Quelltext zusammen und lädt ihn auf die konfigurierte Platine.
	**New:** Erstellt einen neuen Sketch.
	**Open:** Zeigt ein Menü mit allen im Sketchbook gespeicherten Sketches an.
	**Save:** Speichert Ihr Sketch ab.
	**Serial Monitor:** Öffnet den Serial Monitor und erlaubt es Ihnen, die von den Sensoren gesammelten Daten einzusehen.

Zusätzliche Befehle finden Sie in den 5 Menüs: File, Edit, Sketch, Tools und Help.

→ Aufgabe 1: Mach mich an!

Bei dieser Aufgabe sollen die SuS mithilfe von Arduino eine LED steuern. Die SuS sollen verstehen, wie der Quelltext geschrieben werden muss, um die LED blinken zu lassen. Mehr Informationen finden Sie, wenn Sie „Arduino LED blink“ googeln.

Benötigte Komponenten

- 1 Arduino Uno
- 1 Lochrasterplatte
- 1 LED (grün)
- 1 220Ω Widerstand (rot/rot/schwarz/schwarz/braun)
- 2 Kabel

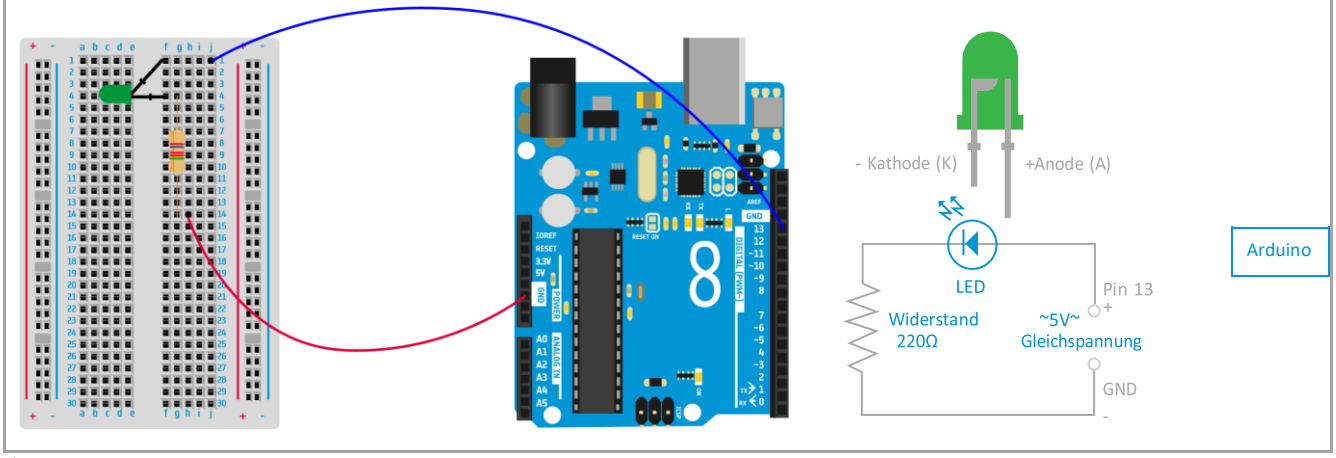
Übung

Aufbau

Der Stromkreis wird wie folgt aufgebaut:

a. Verbinden Sie die grüne LED mit der Lochrasterplatte wie in A5 abgebildet.
b. Verbinden Sie den 220**Ω** Widerstand (rot/rot/schwarz/schwarz/braun) mit dem kurzen Metallstift der LED wie in A5 abgebildet.
c. Verbinden Sie mithilfe eines Kabels den langen Metallstift der LED mit Pin **13** auf dem Arduino Uno.
d. Verbinden Sie mithilfe eines Kabels den Widerstand mit dem **GND** Pin auf dem Arduino Uno.
e. Verbinden Sie schließlich mithilfe des USB-Kabels den Arduino UNO mit einem Computer.

Abbildung A5



↑ Arduino Schaltkreis zum Erleuchten der grünen LED

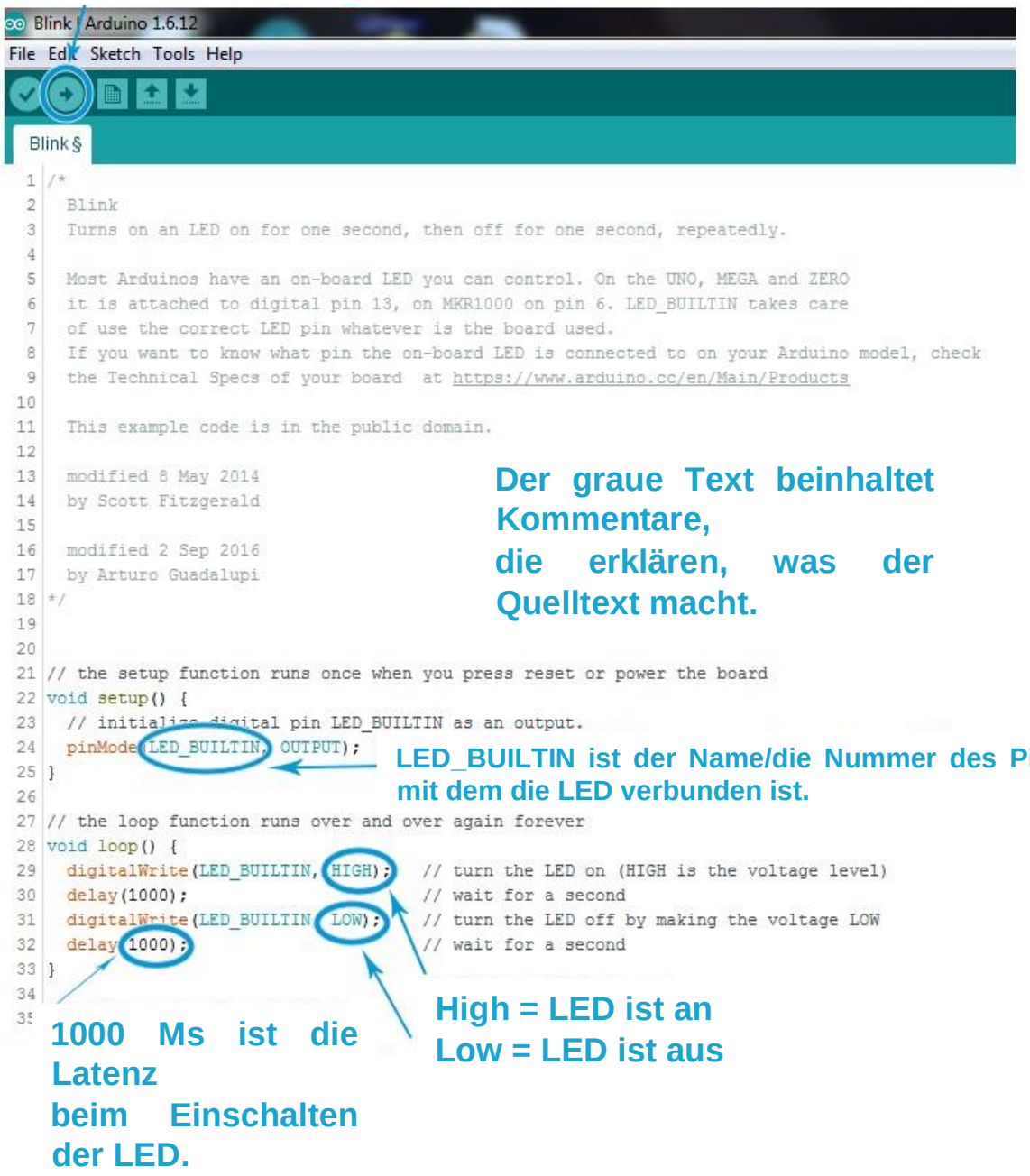
Quelltext

Die SuS sollen nun ein Programm hochladen, mit dem sie die LED kontrollieren können. Sie benutzen einen beispielhaften Quelltext, der wie folgt innerhalb der Arduino Software (IDE) gefunden werden kann. Klicken Sie auf:

File → Examples → Basics → Blink

Der ‚Blink'-Quelltext erscheint auf dem Bildschirm (s. Abb. unten). Kommentare werden in grau nach den Schrägstrichen angezeigt. Sie erklären jede einzelne Zeile des Quelltexts.

Laden Sie das Programm auf Ihren Arduino hoch.



Die hauptsächlichen Bausteine dieses Quelltexts werden nachstehend erklärt:

`void setup()`

Dieser Befehl steht immer am Anfang eines Sketch; er initiiert die Variablen, die im Rest des Sketch verwendet werden. Nach jedem Start oder Reset der Arduino Platine wird dieser Befehl genau einmal ausgeführt.

```
pinMode(LED_BUILTIN,OUTPUT)-> pinMode(13,OUTPUT)
```

Diese Zeile sorgt dafür, dass der angewählte Pin entweder als Input oder als Output fungiert. In dem ‚Blink'-Beispiel kann man **LED_BUILTIN** mit der Nummer des Pins ersetzen, an dem die LED verbunden ist, z.B. **13**.

```
void loop()
```

Dieser Befehl tut genau das, was sein Name suggeriert. Er wiederholt alle Befehle, die er enthält und erlaubt es Ihrem Programm somit mehrere Dinge fortlaufend zu tun.

```
digitalWrite(13,HIGH)
```

Dieser Befehl schaltet die LED ein. HIGH setzt die Spannung bei Pin 13 auf ca. 5V.

```
delay(1000)
```

Dieser Befehl pausiert das Programm für die angezeigte Zeitdauer (Anzeige in MS).

```
digitalWrite(13,LOW)
```

Dieser Befehl schaltet die LED aus, indem er die Spannung auf LOW stellt. LOW bedeutet, dass Pin 13 mit einer Spannung von 0V gespeist wird.

Eine vollständige Liste aller möglichen Befehle finden Sie, wenn Sie ‚Arduino commands' googeln.

→ Aufgabe 2: Signalisiere ein S.O.S.!

Bei dieser Aufgabe verwenden die SuS den Arduino Schaltkreis, den sie in Aufgabe 1 gebaut haben. Sie bedienen die grüne LED und versenden eine Nachricht in Morsecode, um zu demonstrieren, wie sie mit einem Fahrzeug auf dem Mars kommunizieren würden. Um ein ‚O' zu versenden, müssen die SuS die Latenz (delay) im Quellcode ändern, um die Leuchtzeit der grünen LED zu erhöhen. In diesem Fall haben wir die Latenz von 1000 MS zu 5000 MS geändert. Theoretisch funktioniert jede Zeitspanne, die länger ist als 1000 MS. Besagter Quelltext ist abgebildet in Abb. A6:

Abbildung A6

```
void loop() {
  digitalWrite(13, HIGH);
  delay(5000);
  digitalWrite(13, LOW);
  delay(1000);

  digitalWrite(13, HIGH);
  delay(5000);
  digitalWrite(13, LOW);
  delay(1000);

  digitalWrite(13, HIGH);
  delay(5000);
  digitalWrite(13, LOW);
  delay(1000);

}
```

↑ Quelltext um ein ‚o' in Morsecode zu senden

→ Aufgabe 3: Temperaturmessung

Diese Aufgabe besteht darin, dass die SuS mithilfe des Arduino Temperatur messen sollen. Dabei lesen sie die Temperatur vom Temperatursensor (Thermistor*) und erarbeiten sich Verständnis darüber, wie der Quelltext geschrieben werden muss, um den Sensor zu kontrollieren. Mehr Informationen hierzu können Sie finden, wenn Sie ‚Arduino thermistor sensor' googeln.

Komponenten

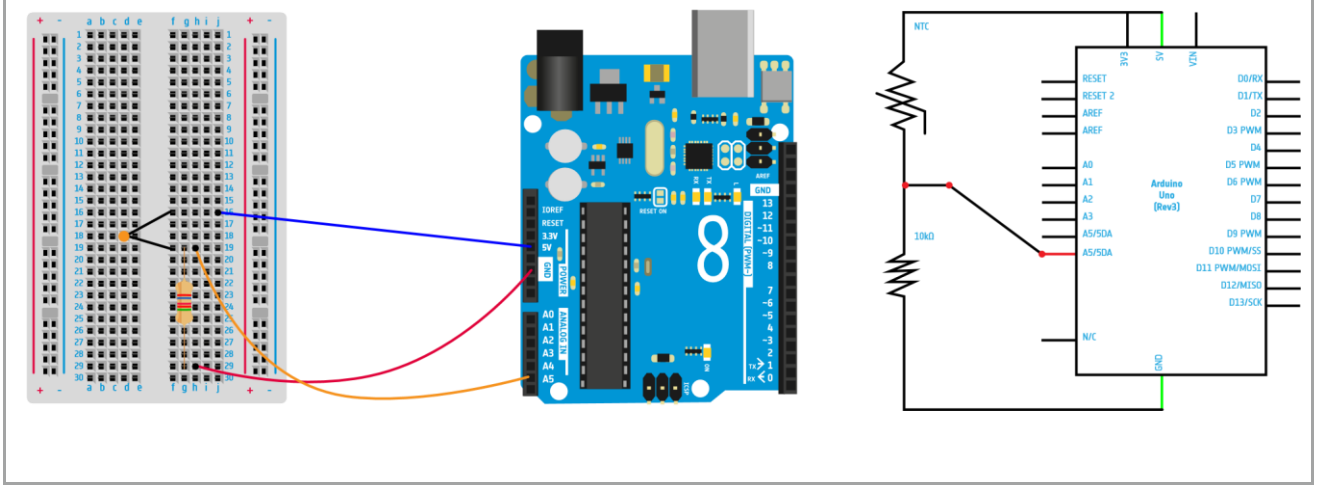
- 1 Arduino Uno
- 1 Lochrasterplatte
- 1 Temperatur Sensor (Thermistor*)
- 1 10kΩ Widerstand (braun/schwarz/schwarz/rot/braun)
- 3 Kabel

Übung

Aufbau

a. Verbinden Sie den Widerstand mit der Lochrasterplatte wie in Abbildung A7.
b. Verbinden Sie den 10kΩ Widerstand (braun/schwarz/schwarz/rot/braun) mit dem Thermistor wie in Abbildung A7.
c. Verbinden Sie den Thermistor mittels eines Kabels mit dem **5V** Pin des Arduino Uno.
d. Verbinden Sie mittels eines Kabels den Widerstand mit dem **GND** Pin des Arduino Uno.
e. Verbinden Sie den Thermistor und den Widerstand mittels eines Kabels mit dem **A5** Pin des Arduino.
f. Verbinden Sie schließlich mittels des USB-Kabels das Arduino Uno mit einem Computer.

Abbildung A7



↑ Arduino Schaltkreis zum Messen von Temperatur

* Thermistor: ein elektrischer Widerstand, dessen Widerstandsfähigkeit durch Hitze beträchtlich reduziert wird

Quelltext

Die SuS sollen nun ihren eigenen Quelltext schreiben. Eine vollständige Liste möglicher Befehle finden Sie, wenn Sie ‚Arduino commands' googeln. Abbildung A8 veranschaulicht einen beispielhaften Quelltext, mit dem Temperatur gemessen werden kann. In diesem Beispiel wird die vom Sensor ermittelte Temperatur als **AnalogT** bezeichnet. Die Maßeinheit ist hier Volt, weil es sich um ein elektrisches Signal handelt; physische Temperatur im eigentlichen Sinne wird hier nicht gemessen. Die Temperatur, die in Grad Celsius konvertiert wird, wird als **AnalogTf** bezeichnet. Der Name des Pins auf dem Arduino Uno ist A5. Sobald der Code hochgeladen wurde und zu laufen beginnt, öffnen Sie das Serial Monitor* Fenster, um zu sehen, wie die Daten gesammelt werden.

Abbildung A8

```
1  void setup() {
2    // put your setup code here, to run once:
3  Serial.begin(9600);
4  }
5
6  void loop() {
7    // put your main code here, to run repeatedly:
8  float AnalogT;
9  float AnalogTf;
10 AnalogT= float(analogRead(A5));
11 AnalogTf = (-40000/AnalogT) + 100;
12 Serial.println();
13 Serial.print("Temperature: ");
14 Serial.print(AnalogTf);
15 Serial.print("C");
16 delay(1000);
17
18 }
```

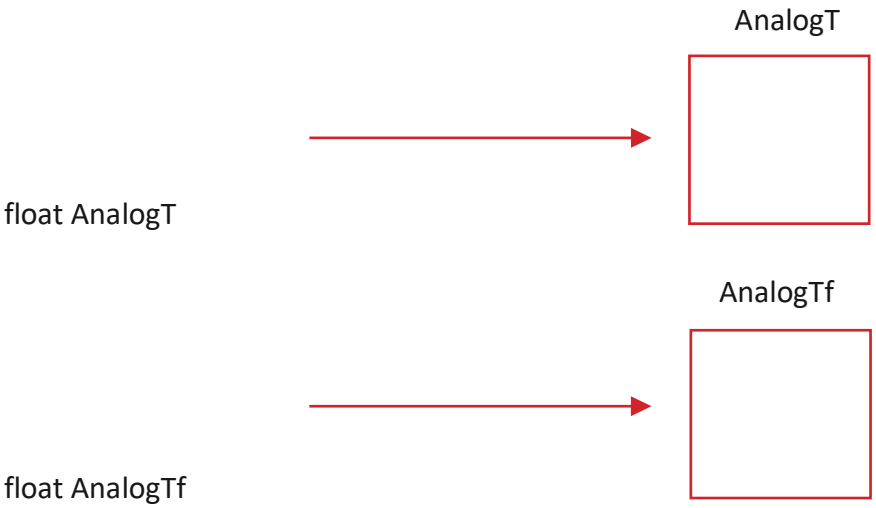
↑ Arduino Quelltext zum Messen von Temperatur

*s. Tabelle A1: Arduino Befehle.

Für Erklärungen des ‚void setup' und des ‚void loop', verweisen Sie auf Aufgabe 1 Übung 3.

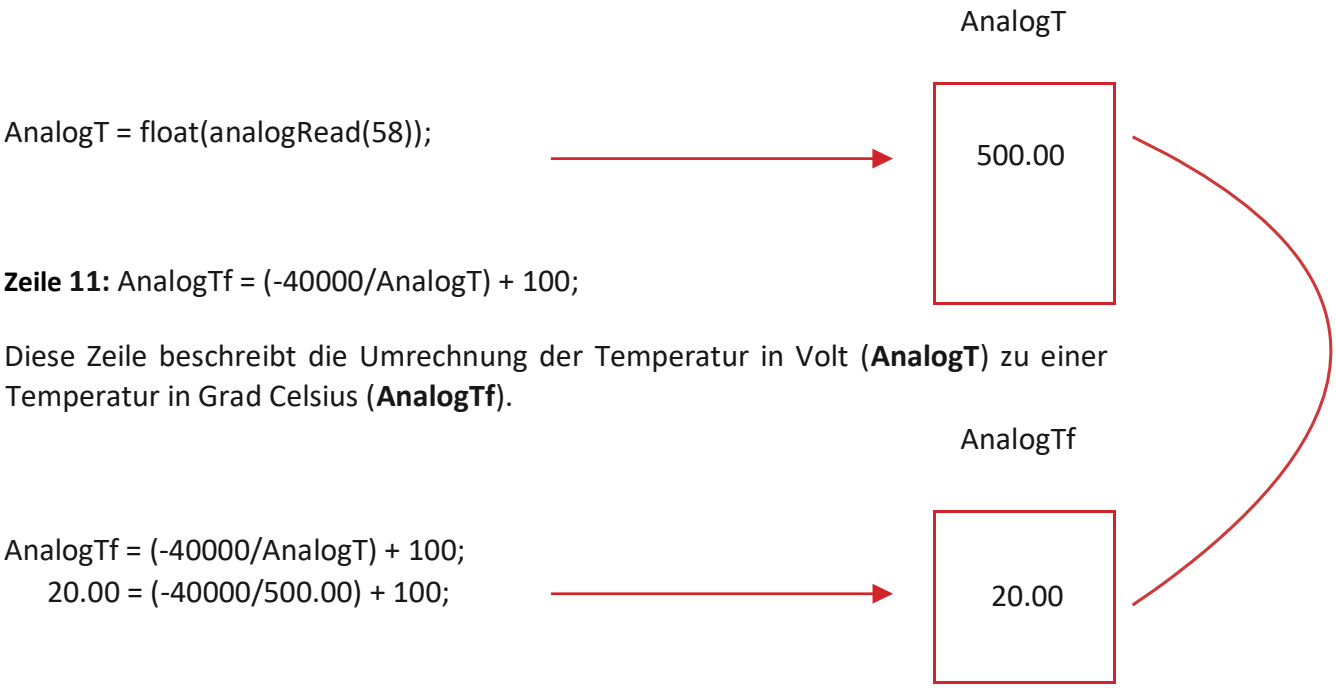
Zeile 3: Da die Temperatur auf dem Computerbildschirm abgebildet wird, ist es notwendig, die Datenrate (9600bits/S) für die Datenübertragung zwischen Arduino und Computer vorzugeben.

Zeilen 8-9: Diese beiden Quelltextzeilen werden dazu verwendet, die Variablen AnalogT und AnalogTf zu definieren. Die Bezeichnungen werden vom Programmierer festgelegt. Um diesen Prozess modelhaft darzustellen, verwenden wir eine grafische Erklärungshilfe. Wenn Sie ‚float AnalogT' schreiben, erstellen Sie innerhalb des Computers eine Box oder einen leeren Raum, der als AnalogT bezeichnet wird. Dieser hat eine spezifische Größe, die von ‚float' angezeigt wird. ‚Float' bedeutet, dass die Zahl innerhalb der Box ein Dezimalkomma hat.



Zeile 10: AnalogT = float(analogRead(A5));

Der Computer schreibt die Zahl, die er am A5 Pin des Arduino abliest (z.B. 500.00), in die AnalogT-Box. Wir verwenden die Funktion **analogRead()**, um jeglichen Pin des Arduino abzulesen.



Zeile 11: AnalogTf = (-40000/AnalogT) + 100;

Diese Zeile beschreibt die Umrechnung der Temperatur in Volt (**AnalogT**) zu einer Temperatur in Grad Celsius (**AnalogTf**).

Zeilen 12-15: Um die Temperaturmessung auf dem Bildschirm darzustellen, müssen Sie die ‚print'-Funktion verwenden. Diese Funktion wird Zahlen innerhalb jedweder Box wiederherstellen und diese dann auf dem Serial Monitor abbilden. In unserem Beispiel wird die ‚print'-Funktion die Zahl in der Box mit der Bezeichnung AnalogTf abbilden. Alle Wörter innerhalb von "…" werden direkt auf dem Serial Monitor abgebildet.

Zeile 16: Die ‚delay'-Funktion wird dazu verwendet, 1000 MS zu warten und dann die Temperatur auf dem Serial Monitor abzubilden.

Diskussion

Die SuS werden ihre eigenen Temperatursensoren wahrscheinlich kalibrieren müssen. Beim Kalibrieren können sie als Referenz ein weiteres Thermometer verwenden. Um grob zu kalibrieren, führen die SuS den Quelltext so aus, wie sie ihn geschrieben haben und wenn die Messergebnisse nicht der Referenz-temperatur entsprechen, können sie die letzte Nummer (+100) in Zeile 11 (AnalogTf) (Abbildung A8) anpassen.

Praxisübung

Die SuS sollen den Temperatursensor so kalibrieren, als würde er sich auf der Oberfläche des Mars befinden. Da die Durchschnittstemperatur des Mars -60 °C beträgt und die der Erde 14 °C, müssen die SuS die Differenz von -74 miteinkalkulieren. Der Quelltext muss hier wie folgt angepasst werden:

AnalogTf = (-40000/AnalogT) + 100 – 74;

→ Aufgabe 4: Druckmessung

An dieser Stelle sollen die SuS den Umgebungsdruck mithilfe des Arduino messen. Die SuS lernen den Drucksensor (MPX4115A) kennen und begreifen, wie der Quelltext geschrieben werden muss, um den Sensor steuern zu können. Mehr Informationen hierzu können Sie finden, wenn Sie ‚Arduino MPX4115A' googeln:

Komponenten

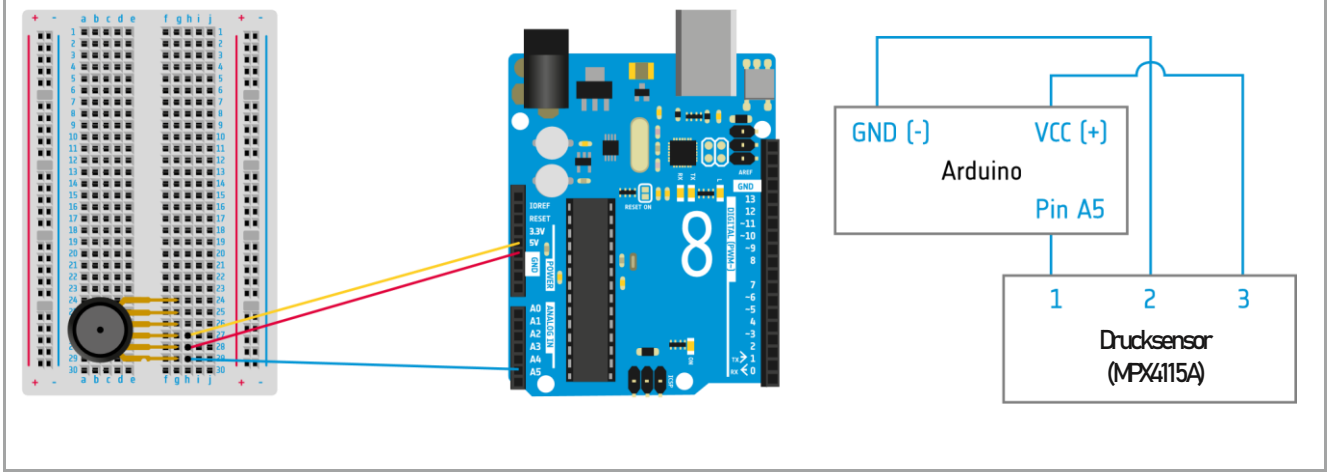
- 1 Arduino Uno
- 1 Lochrasterplatte
- 1 Drucksensor
- 3 Kabel

Übung

Aufbau

a. Verbinden Sie den Drucksensor mit der Lochrasterplatte.
b. Verbinden Sie mithilfe eines Kabels Metallstift 1 des Drucksensors (hat eine Kerbe) mit Pin **A5**.
c. Verbinden Sie mithilfe eines Kabels Metallstift 2 des Drucksensors mit der **GND** Reihe des Arduino.
d. Verbinden Sie mithilfe eines Kabels Metallstift 3 mit dem **5V** Pin des Arduino.
e. Verbinden Sie schließlich mithilfe eines USB Kabels den Arduino mit einem Computer.

Abbildung A9



↑ Arduino Schaltkreis zum Messen von Druck

Quelltext

In dieser Aufgabe verwenden die SuS dieselben Befehle wie schon in der Temperaturmessungsaufgabe; erneut konzentrieren sie sich auf die Kalibrierung eines Sensors. Ein wichtiger und sehr sinnvoller Schritt bei der Verwendung eines Sensors ist es, sich dessen Datenblatt einmal genauer anzuschauen. Dieses enthält nämlich detaillierte Informationen darüber, wie der Sensor genau funktioniert. In diesem Fall können wir schnell nützliche Informationen erhalten, wenn wir ‚MPX4115A datasheet' googeln.

Der Drucksensor konvertiert den gemessenen Druck in elektrische Spannung. Dabei handelt es sich um ein analoges Signal, welches in ein digitales übersetzt wird. Um die gemessenen Spannungen in Umgebungsdruckwerte zu übersetzen (in die Einheit Pa), benötigen wir die Transferfunktion des Sensors. Diese Funktion beschreibt die mathematische Beziehung zwischen der Ausgangsspannung des Sensors und dem entsprechenden Druck. Diese Funktion kann auf dem Datenblatt des Sensors nachgelesen werden. Sie kann von Sensor zu Sensor unterschiedlich sein.

Wenn wir uns das Datenblatt anschauen, finden wir die folgende Transferfunktion: $\mathbf{V_{out} = V_S (P \times 0.009 - 0.095)}$. Wir müssen lediglich den Wert von P (Druck) isolieren und bedenken, dass V_S = 1024 ist (die abgelesene Spannung wird in 1024 Schritte aufgeteilt). $\mathbf{V_{out}}$ = Analoger Wert, den wir von Pin **A5** ablesen.

Das nachstehende Beispiel ist ein Quelltext der Druckwerte darstellt. Er kann als Referenz verwendet werden.

Abbildung A10

```
1  void setup() {
2    // put your setup code here, to run once:
3     Serial.begin(9600);
4  }
5
6  void loop() {
7    // put your main code here, to run repeatedly:
8     float AnalogP;
9     float AnalogPf;
10    AnalogP= float(analogRead(A5));
11    AnalogPf = ((AnalogP/1024)+0.095)/0.009;
12    Serial.println();
13    Serial.print("Pressure: ");
14    Serial.print(AnalogPf);
15    Serial.print("kPa");
16    delay(1000);
17
18 }
```

↑ Arduino Quelltext zum Messen von Druck

Die Bezeichnung für den Druck in Volt ist AnalogP, Druck in Pascal wird hier als AnalogPf dargestellt. Die Bezeichnung für den Pin, an dem der Drucksensor angeschlossen ist, lautet A9.

Um den Drucksensor zu kalibrieren, können die SuS den gegenwärtigen Umgebungsdruck im Internet nachschauen. Für eine grobe Kalibrierung können sie den Quelltext so wie er ist ausführen und wenn die Ergebnisse abweichen sollten, können sie ihn in Zeile 11 (**AnalogPf**) angleichen.

→ Aufgabe 5: Höhenmessung

Druck variiert je nach Höhe und somit kann er dazu verwendet werden, Höhen zu messen. Um Höhen messen zu können, müssen die SuS den folgenden Quelltext in den Loop des Quelltexts einbauen, der den Druck misst.

Es folgt ein Beispiel für einen Quelltext, der die Höhe in Metern anzeigt. Dieser kann als Referenz benutzt werden.

Abbildung A11

```
float Altitude;
float Altitudef;
  Altitude = pow(AnalogPf/101.325,0.1903);
  Altitudef = (1-Altitude)*44300 + 100;
Serial.print("Altitude: ");
Serial.print(Altitudef); Serial.print ("meters");
```

↑ Arduino Quelltext zum Messen von Druck

Die Bezeichnung für die angezeigte Höhe in Metern ist Altitudef. Der Druck, gemessen in kPa, wird in diesem Beispiel von AnalogPf angezeigt.

Aufgrund der Hinweise aus der ‚Wusstest du schon …?' Box sollten die SuS herleiten können, dass die Temperatur der Atmosphäre fällt, je höher man aufsteigt und diese Beziehung sollte aus den von den SuS erstellten Graphen bzgl. Höhe und Temperatur ersichtlich werden.

Wie man Arduino im Klassenraum verwendet

Arduino ist ein hervorragendes Tool, um SuS die Grundlagen des Programmierens sowie das große Spektrum der damit verbundenen Anwendungsmöglichkeiten zu vermitteln. Es kann auch dazu verwendet werden, MINT-Fächer zu lehren und zu lernen. Beispiele hierfür sind:

- **Einführung in die Grundlagen von Schaltkreisen** - Aufgabe 1 kann dazu benutzt werden, den SuS einen einfachen Stromkreis zu erklären.
- **Einführung in die Datenanalyse** – SuS sammeln Daten aus der realen Welt (Temperatur und/oder Druck) und analysieren diese Daten mithilfe von Graphen. Die SuS lesen Informationen ab und ziehen Schlussfolgerungen.
- **Wetterelemente und wie man Daten sammelt** – Die SuS können ihre eigene Wetterstation entwickeln und sonnige oder wolkige Tage vorhersagen.

→ LINKS

Arduino Blog für die neuesten Informationen rund um Arduino: https://blog.arduino.cc/

Anleitungen für die Verwendung unterschiedlicher Arduino Komponenten: https://quarkstream.wordpress.com/

Viele Arduino basierte Projekte für zuhause findet man auf: http://www.instructables.com/howto/arduino/

Auf HackADay findet man weitere Arduino basierte Projekte: https://hackaday.io/list/3611-arduino-projects