I.D.35

Elektrizitätslehre und Magnetismus

Vom Blinklicht zur Ampelschaltung – Programmieren mit dem Mikrocontroller Arduino

Thomas Rosenthal, Esslingen am Neckar
Illustrationen von: Dr. Wolfgang Zettlmeier

© Spyderskidoo/iStock/Getty Images Plus

In vielen technischen Geräten sind heutzutage Mikrocontroller verbaut. Ihre Schüler sollen dies am Beispiel der Programmierung einer Ampelschaltung an einer Kreuzung erkennen. Dabei können sie bei der Autofahrerampel eine Tag-und-Nacht-Situation simulieren, während bei der Fußgängerampel per Taster die Farbe Grün, begleitet von einem Ton, angefordert werden kann. Dabei lernen sie auch elektronische Bauteile und deren Zusammenführung zu elektronischen Schaltungen kennen.

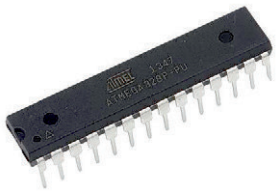
KOMPETENZPROFIL

Klassenstufe/Lernjahr:	9/10
Dauer:	10 Unterrichtsstunden
Kompetenzen:	1. Fachwissen: Mikrocontroller kennenlernen; 2. Erkenntnisgewinnung: einzelne Bauteile programmieren und in einem Projekt zusammenführen; 3. Methodik: elektronische Schaltungen stecken und Schaltpläne erstellen
Thematische Bereiche:	elektronische Schaltungen, Programmierung
Medien:	Arbeitsblätter, Farbfolie, Programmcodes
Zusatzmaterialien:	Lösungen und Erweiterungsmöglichkeiten zur Differenzierung

Sachanalyse

Zum Mikrocontroller im Allgemeinen und zum Arduino im Besonderen

Als Mikrocontroller (auch µController, µC) bezeichnet man **Halbleiterchips**, die neben einem Prozessor zugleich auch Peripheriefunktionen besitzen. In vielen Fällen befindet sich auch der Arbeits- und Programmspeicher teilweise oder komplett auf demselben Chip. Ein Mikrocontroller ist ein Ein-Chip-Computersystem und hat deshalb eher eine geringe Leistungsfähigkeit. Er muss oft auch nur bestimmte, wenige Prozesse ausführen. Diese funktionieren nach dem EVA-Prinzip: Eingabe – Verarbeitung – Ausgabe:



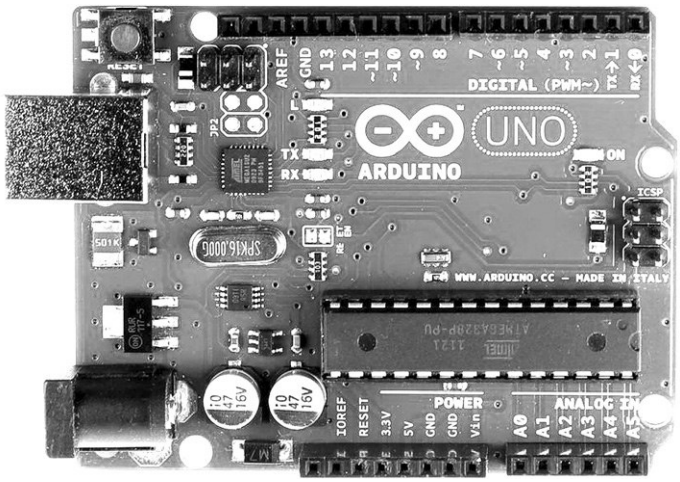
Quelle: de.rs-online.com
© Arduino www.arduino.cc

1. **Eingabe**: Damit eine Datenverarbeitung überhaupt stattfinden kann, müssen zunächst einmal Daten vorhanden sein. Diese lassen sich über Tastatur, Maus, Gamepad, Scanner, Mikrofon oder Webcam in das Computersystem eingeben.
2. **Verarbeitung**: Nach der Eingabe von Daten in ein Computersystem kann die Recheneinheit (CPU, Prozessor, Controller) darauf zugreifen. Die CPU, die aus Speicher, Steuer- und Rechenwerk besteht, berechnet aus der Dateneingabe die Datenausgabe. Für die nötige Berechnung oder zur späteren Aufbewahrung werden die Daten (zwischen-)gespeichert. Die gängigsten Speicher sind: Festplatte, SSD, Arbeitsspeicher (RAM), ROM, CD, DVD, SD-Karte oder USB-Stick.
3. **Ausgabe**: Damit die berechneten Daten nun zur Verfügung stehen, müssen sie in einer bestimmten – der jeweils gewünschten – Form wieder ausgegeben werden. Dies erfolgt am häufigsten durch Bildschirm, Drucker, Lautsprecher oder Beamer.

Beim Arduino handelt es sich um eine aus Soft- und Hardware bestehende **Physical-Computing-Plattform.**

Im Sinne von Open Source sind beide Komponenten quelloffen. Dabei besteht die Hardware aus einem einfachen Eingabe-/Ausgabe(EA)-Board mit einem Mikrocontroller, der analoge und digitale Ein- und Ausgänge hat. Die kostenlose Software als zugehörige Entwicklungsumgebung basiert auf Processing und erleichtert auch technisch weniger Versierten den Zugang zur Programmierung im Allgemeinen und zu Mikrocontrollern im Besonderen.

Die Programmierung selbst erfolgt in einer C bzw. C++ ähnlichen Programmiersprache; umfangreiche Bibliotheken und Beispiele vereinfachen die Programmierung erheblich. Das erste Board wurde im Jahre 2005 von Massimo Banzi und David Cuartielles entwickelt. Der Name „Arduino" stammt von der Bar in Ivrea (Italien), in der sich einige der Projektgründer oft trafen.



© Arduino www.arduino.cc

Auf einen Blick

1. Stunde (Klassenzimmer)

Thema: **Mikrocontroller im Alltag – Farbfolie**

M 1 (Fo) **Mikrocontroller im Alltag** / Gedankenaustausch über technische Geräte und ihre Funktionsweisen, die sich auf den Einsatz von Mikrocontrollern beziehen.

M 2 (Tx) **Mikrocontroller im Alltag** / Austausch über die Funktionsweise eines Mikrocontrollers an einem praktischen Beispiel und Verallgemeinerung auf das EVA-Prinzip.

Benötigt:
- ☐ OH-Projektor / Beamer / Visualiser / Whiteboard
- ☐ Arduino-Boxen und ggf. USB-Sticks
- ☐ mehrfach ausgedrucktes und entsprechend der Schülerzahl zugeschnittenes Word-Dokument „Arduino – Organisation"

2./3. Stunde (Computerraum)

Thema: **Aufbau und Funktionsweise des Arduino /** Die Arduino-Software umgebung

M 3 (Tx) **Aufbau und Funktionsweise des Arduino-Boards** / Kennenlernen und Verständnis für den Aufbau und die Funktionsweise des Arduino-Boards

M 4 (Tx) **Die Arduino-Softwareumgebung – Teil 1** / Kennenlernen und Verständnis für den Aufbau und die Funktionsweise der Arduino-Softwareumgebung

M 5 (Tx) **Die Arduino-Softwareumgebung – Teil 2** / Kennenlernen und Verständnis für den Aufbau und die Funktionsweise der Arduino-Softwareumgebung

M 6 (Tx) **Ein erstes Blinken einer LED** / Kennenlernen und Verständnis für den Aufbau eines Programmcodes (Sketches) in der Arduino-Softwareumgebung

Benötigt:
- ☐ OH-Projektor / Beamer / Visualiser / Whiteboard
- ☐ digitale Fassung von Schaltplan „01_Blinkende_LED"
- ☐ Arduino-Boxen und USB-Sticks
- ☐ 1 rote LED für jeden Schüler

4./5. Stunde (Computerraum)

Thema: **Verwendung eines Steckbrettes und Autofahrerampel** / Verwendung von Variablen und Ampelkreuzung

M 7 (Tx) **Verwendung eines Steckbrettes und Autofahrerampel** / Kennenlernen des Steckbrettes, Schaltung und Programmierung einer Ampelschaltung

M 8 (Tx) **Verwendung von Variablen und Fußgängerampel** / Kennenlernen der Vereinfachung eines Programmes unter Verwendung von Variablen, Schaltung und Programmierung einer Fußgängerampel

Benötigt:
- ☐ OH-Projektor / Beamer / Visualiser / Whiteboard
- ☐ digitale Vorlage des Word-Dokumentes „Zur Geschichte der Ampel“
- ☐ Kopie oder digitale Vorlage des Word-Dokumentes „AB Widerstände“
- ☐ digitale Fassungen von Schaltplänen „02_Autofahrer-Ampel“ und „04_ Ampelkreuzung“
- ☐ Arduino-Boxen und USB-Sticks
- ☐ 2 rote, 1 gelbe, 2 grüne LEDs, fünf 220 Ω-Widerstände, 11 (5 kürzere und 6 längere) Jumperkabel für jeden Schüler

6./7. Stunde (Computerraum)

Thema: **Töne mit dem Lautsprecher**

M 9 (Tx) **Töne mit dem Lautsprecher abspielen – Teil 1** / Kennenlernen der Funktionsweise eines Lautsprechers und des zugehörigen Befehles

M 10 (Tx) **Töne mit dem Lautsprecher abspielen – Teil 2** / Programmierung von Tönen, einer Melodie und Erweiterung der Fußgängerampel um einen Blindenton

Benötigt:
- ☐ OH-Projektor / Beamer / Visualiser / Whiteboard
- ☐ digitale Vorlage des Word-Dokumentes „Blindenampel“
- ☐ digitale Fassungen von Schaltplänen „05_Lautsprecher“ und „06_Ampelkreuzung_Ton“
- ☐ Arduino-Boxen und USB-Sticks
- ☐ 2 rote, 1 gelbe, 2 grüne LEDs, fünf 220 Ω-Widerstände, 13 (6 kürzere und 7 längere) Jumperkabel, 1 Piezometer für jeden Schüler

8./9./10. Stunde (Computerraum)

Thema: **Die Ampel bei Tag und Nacht / Auf Knopfdruck wird es grün**

M 11 (Tx) **Die Ampel bei Tag und Nacht** / Kennenlernen der Funktionsweise eines lichtabhängigen Widerstandes und der zugehörigen Programmierung

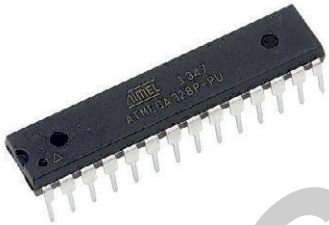
M 12 (Tx) **Auf Knopfdruck wird es grün** / Kennenlernen der Funktionsweise eines Tasters und der gehörigen Programmierung

Benötigt:
- ☐ OH-Projektor / Beamer / Visualiser / Whiteboard
- ☐ Kopie und/oder digitale Vorlagen der Word-Dokumente „AB Serieller Monitor“ und, „AB Spannungsteiler“
- ☐ digitale Fassungen von Schaltplänen „07_Helligkeit_seriell“, „08_Nachtampel“ und „09_Ampelkreuzung_Tag_und_Nacht“
- ☐ digitale Fassungen von Schaltplänen „10_Taster_Unterprogramm“ und „11_Ampelkreuzung_Tag_und_Nacht_Taster“
- ☐ Arduino-Boxen und USB-Sticks
- ☐ 2 rote, 1 gelbe, 2 grüne LEDs, zwei 10–kΩ – und fünf 220–Ω-Widerstände, 20 (10 kürzere und 10 längere) Jumperkabel, 1 Taster, 1 Piezometer für jeden Schüler

M 1 Mikrocontroller im Alltag – Farbfolie

© Martin_Poole / Digital Vision / Getty Images Plus

© grinvalds / iStock / Getty Images Plus



Quelle: de.rs-online.com
© Arduino www.arduino.cc

Mikrocontroller (auch µController, µC) sind Halbleiterchips, die einen Prozessor und sogenannte Peripheriefunktionen enthalten. Ein Mikrocontroller ist ein Ein-Chip-Computersystem.

© Yuri_Arcurs / E+ / Getty Images Plus

© Guido Mieth / Digital Vision / Getty Images Plus

Aufgaben

1. Beschreibe Funktionen der einzelnen abgebildeten Geräte.
2. Gibt es technische Gemeinsamkeiten?
3. Kennst du weitere Geräte und kannst du deren Funktionsweise beschreiben?

Mikrocontroller im Alltag

M 2

Viele Autofahrer besitzen heutzutage einen Autoschlüssel, der per Tastendruck die Autotür öffnet. In früheren Zeiten geschah dies nach dem Schlüssel-Schloss-Prinzip einer normalen Eingangstür, indem ein Schlüssel in einen Schlitz an der Autotür gesteckt werden musste.
Doch wie funktioniert eigentlich ein solcher moderner Autoschlüssel?
Sicherlich steckt in ihm eine ganze Menge Elektronik; doch ein Computer hätte darin keinen Platz und wäre auch viel zu teuer und zu „komplex" für eine solche relativ einfache Steuerung. Man verwendet dabei sogenannte **Mikrocontroller** (auch µController, µC). Dabei handelt es sich um Halbleiterchips, die einen Prozessor und sogenannte Peripheriefunktionen enthalten. Ein Mikrocontroller ist ein Ein-Chip-Computersystem, also ein Computer, jedoch mit viel geringerer Leistungsfähigkeit. Mikrocontroller vereinen verschiedene Bauteile wie Recheneinheit (CPU), Speicher, Schnittstellen wie USB, Display-Controller und Analog-Digital-Wandler auf einem einzigen Chip. Der Vorteil von Mikrocontrollern ist, dass sie speziell für ihre Aufgabe konzipiert werden können und relativ günstig sind.
Mikrocontroller funktionieren nach dem sogenannten EVA-Prinzip.

© republica / E+ / Getty Images Plus

Das EVA-Prinzip: Eingabe – Verarbeitung – Ausgabe:
Ihr kennt dieses Prinzip bei der Eingabe eines Textes am PC oder am Handy:
Einzelne Buchstaben werden über die Tastatur eingegeben. Die CPU des Computers verarbeitet die Tastaturanschläge (Position, Farbe, Größe …) anschließend. Euer Text wird als Ergebnis auf dem Monitor im Textprogramm ausgegeben.

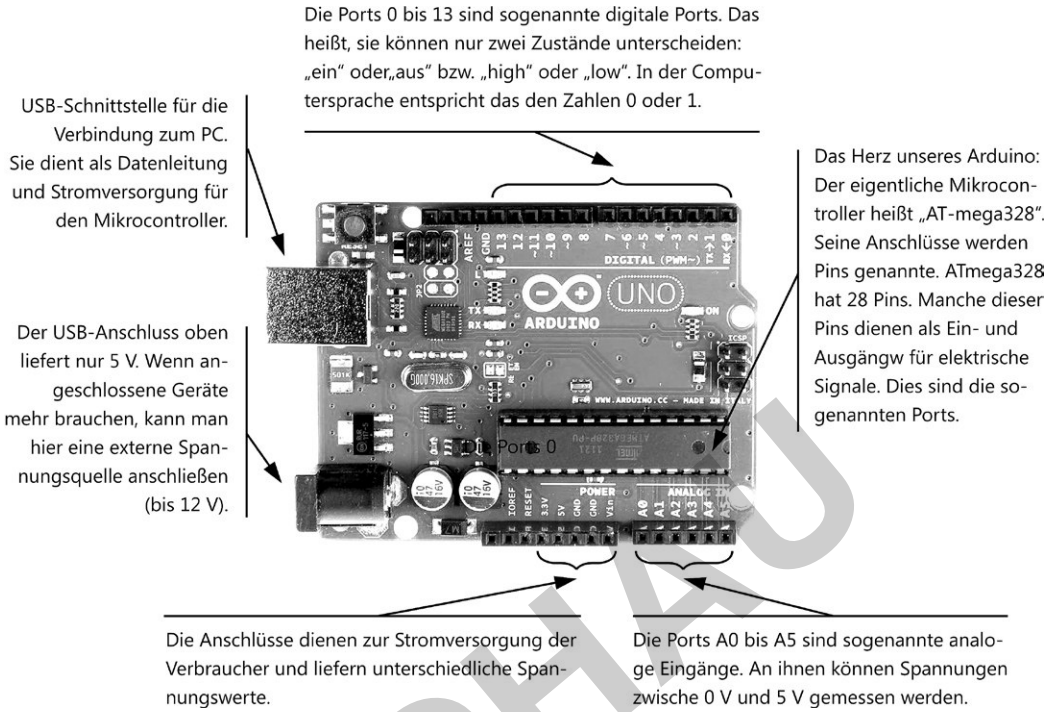
1. **Eingabe:** Damit eine Datenverarbeitung überhaupt stattfinden kann, müssen zunächst einmal Daten vorhanden sein. Diese lassen sich über Tastatur, Maus, Gamepad, Scanner, Mikrofon oder Webcam in das Computersystem eingeben.
2. **Verarbeitung:** Nach der Eingabe von Daten in ein Computersystem kann die Recheneinheit (CPU, Prozessor, Controller) darauf zugreifen. Die CPU, die aus Speicher, Steuer- und Rechenwerk besteht, berechnet aus der Dateneingabe die Datenausgabe. Für die nötige Berechnung oder zur späteren Aufbewahrung werden die Daten (zwischen-)gespeichert. Die gängigsten Speicher sind: Festplatte, SSD, Arbeitsspeicher (RAM), ROM, CD, DVD, SD-Karte oder USB-Stick.
3. **Ausgabe:** Damit die berechneten Daten nun zur Verfügung stehen, müssen sie in einer bestimmten – der jeweils gewünschten – Form wieder ausgegeben werden. Dies erfolgt am häufigsten durch Bildschirm, Drucker, Lautsprecher oder Beamer.

Aufgaben

1. Beschreibe die Funktionsweise eines Autoschlüssels.
2. Übertrage das EVA-Prinzip auf den Autoschlüssel.

M 3 Aufbau und Funktionsweise des Arduino-Boards

Nun lernst du das Arduino-Uno-Board, mit dem du in Zukunft arbeiten wirst, genauer kennen.



© *Arduino www.arduino.cc*

Als **Hardware** bezeichnet man im Allgemeinen die mechanischen und elektronischen Bauteile eines Mikrocontrollers bzw. PCs – also im Prinzip alles, was man „in die Hand nehmen“ kann. Der Begriff „hardware“ kommt aus dem Englischen und bedeutet übersetzt ursprünglich „Eisenwaren“. Im englischsprachigen Raum besitzt der Begriff diese Bedeutung auch heute noch. Zusätzlich hat er sich aber weltweit als Überbegriff für Bauteile eines Computersystems verbreitet. Auf der Platine des Arduinos sind neben dem eigentlichen Mikrocontroller noch so manche kleine Bauteile wie zum Beispiel Widerstände, Spannungswandler oder Strombegrenzer verbaut und viele verschiedene Anschlüsse angebracht.

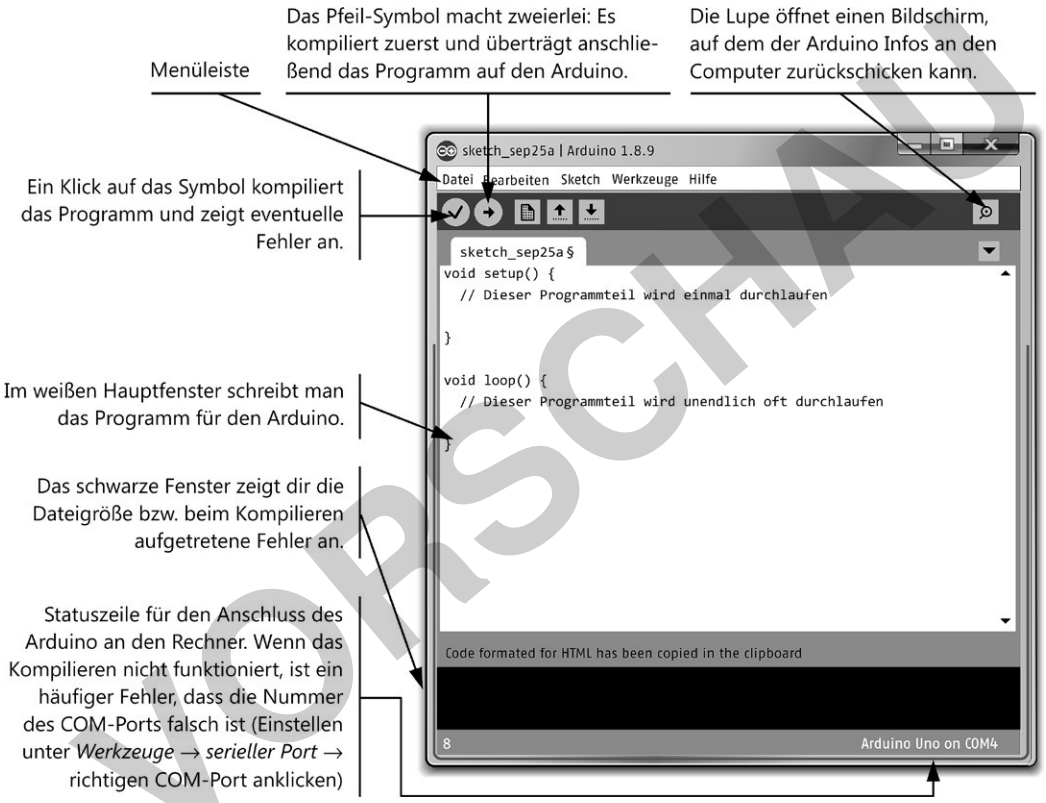
Arduino versus Handy: Bei Computern oder Handys wird oft die Taktfrequenz des Prozessors angegeben. Damit ist die Geschwindigkeit gemeint, mit der Daten verarbeitet werden können. Der Atmega328 des Arduinos besitzt eine Taktfrequenz von 16 MHz und hat insgesamt 32 Kilobyte Speicherplatz. Ein Vergleich mit dem iPhone 8 lässt den Arduino dagegen „ziemlich alt“ aussehen: Es hat einen Prozessor mit 1,8 GHz, also 1800 MHz, Taktfrequenz und 3 GB, also 3.000.000 Kilobyte, Arbeitsspeicher. Beachte jedoch: Beim Flug zum Mond half Neil Armstrong ein Computer mit 4 Kilobyte Arbeitsspeicher und einem 1-MHz-Prozessor.

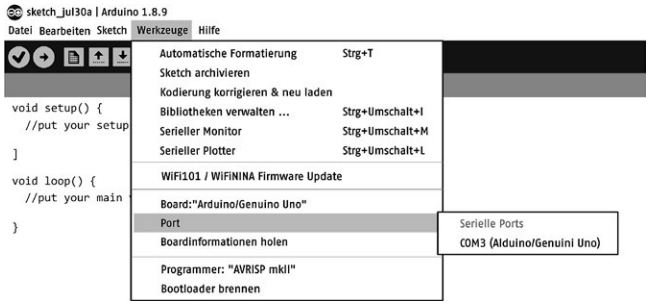
Aufgaben

1. Beschreibe den Aufbau des Arduino-Boards.
2. Problematisiere die Leistungsfähigkeit eines Mikrocontrollers.

Die Arduino-Softwareumgebung – Teil 1 M 4

Um den Arduino in seiner Funktion als Mikrocontroller nutzen zu können, ist er mit dem PC zu verbinden. Streng gesehen ist der Arduino jedoch nicht sehr „intelligent“, da er nicht eigenständig denken kann; vielmehr führt er nur zuvor über eine Software eingegebene Befehle der Reihe nach aus. Eine solche Abfolge von Befehlen heißt **Programm** bzw. **Sketch**. Programmiert wird der Arduino in einer sogenannten Entwicklungsumgebung. Dort schreibt man das Programm für den Arduino in der entsprechenden Programmiersprache. Diese ist so ähnlich wie die berühmte Programmiersprache „C“. Aber der Arduino spricht noch eine andere Sprache, denn bevor das geschriebene Programm über das USB-Kabel auf den Arduino übertragen wird, übersetzt es ein elektronischer Dolmetscher in Maschinensprache. Diesen Vorgang nennt man **Kompilieren.**
Öffne auf dem Desktop durch Doppelklicken mit der Maus die Arduino-Software und verwende den Vollbildschirmmodus des Programmes durch einen Klick auf das Kästchen rechts oben.





Verbinde den Arduino nun über das USB-Kabel mit dem PC. Prüfe anschließend über **Werkzeuge → Port**, ob der Arduino mit dem PC richtig verbunden ist. Sieht es so wie in der nebenstehenden Abbildung aus, ist „COM3 (Arduino/Genuino Uno)“ durch einen Klick mit der linken Maustaste zu aktivieren. Es erscheint ein blauer Haken davor. Wiederholt man den Schritt Werkzeuge → Port, so steht nun hinter Port der jeweilige Arduino.

PORT: “COM3(Arduino/Genuino Uno)”>

Verwendung von Variablen und Fußgängerampel

M 8

Variablen können ein umfangreiches Programm wesentlich vereinfachen und tragen dazu bei, bei vielen angeschlossenen Komponenten die Übersicht zu behalten. Verschiedenen Anschlüssen können über Variablen konkrete Namen zugeordnet werden, sodass im späteren Programm sehr einfach diese Namen verwendet und vor allem wiedererkannt werden können. Wenn man später mehrere Bauteile gleichzeitig einsetzt, behält man einen besseren Überblick, weil man keine Zahlen wie zum Beispiel die Ports des Arduinos, sondern lediglich die zuvor zugeordneten Namen der entsprechenden Bauteile verwendet.

Tipp:

Bei der Namensgebung dürfen keine Umlaute oder Sonderzeichen verwendet werden.

Beispiel eines Programmes mit zwei LEDs:

Ohne Variablen

```
/*
Abwechselndes Blinken
Rote LED an Port 13, Grüne LED an
Port 12
*/

void setup()
{
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  digitalWrite(12, LOW);
  delay(1000);
  digitalWrite(13, LOW);
  digitalWrite(12, HIGH);
  delay(1000);
}
```

Mit Variablen

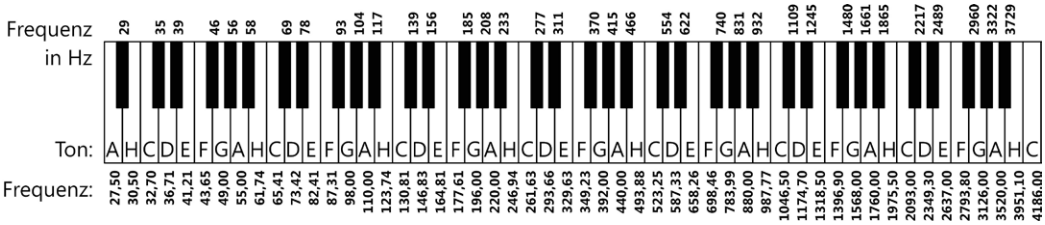
```
/*
Abwechselndes Blinken
Rote LED an Port 13, Grüne LED an
Port 12
*/

int ledrot = 13;
int ledgruen = 12;

void setup()
{
  pinMode(ledrot, OUTPUT);
  pinMode(ledgruen, OUTPUT);
}

void loop()
{
  digitalWrite(ledrot, HIGH);
  digitalWrite(ledgruen, LOW);
  delay(1000);
  digitalWrite(ledrot, LOW);
  digitalWrite(ledgruen, HIGH);
  delay(1000);
}
```


Aufgaben

1. Vergleiche den Sketch „Ohne Variablen“ mit dem Sketch „Mit Variablen“: Wie viele Befehle musst du jeweils umschreiben, wenn du die LEDs statt an den Ports 12 und 13 an den Ports 10 und 11 anschließen willst?
2. Schreibe dein Programm der Autofahrerampel so um, dass du für die drei LEDs Variablen verwendest. Hast du eine Idee für die Einführung einer weiteren Variablen?
3. Speichere den zugehörigen Sketch unter „03_Autofahrer-Ampel_Variable“.
4. Ergänze den int-Befehl in deinem Glossar.
5. Erweitere dein Programm der Autofahrerampel um eine Fußgängerampel. Beachte dabei den korrekten Ablauf der einzelnen Ampelfarben und gib sinnvolle Zeiten ein.
6. Speichere den zugehörigen Sketch unter „04_Ampelkreuzung“.

Töne mit dem Lautsprecher abspielen – Teil 2

M 10



Tipp:

Wie du vielleicht schon bemerkt hast, wartet das Programm nach einem *tone*-Befehl nicht, bis der gesamte Ton abgespielt ist, sondern bearbeitet sofort die nächste Programmzeile. Deshalb ist immer ein *delay*-Befehl notwendig, um sicherzustellen, dass ein Ton auch wirklich in der ganzen Länge abgespielt wird. Es gibt auch einen Befehl, um die Tonausgabe abzuschalten: noTone(Port);

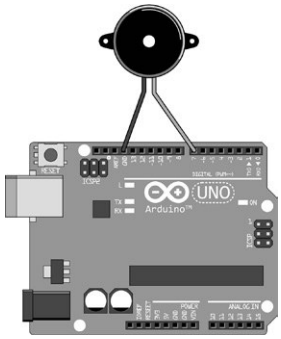
Lebewesen	Hörumfang in Hz
Mensch	16–20 000
Hund	15–50 000
Katze	60–65 000
Delfin	150–150 000
Fledermaus	1000–120 000

```
void setup() {
 pinMode (7,OUTPUT);
}

void loop() {
tone(7,220,1000);
delay(1000);
tone(7,440,1000);
delay(1000);
tone(7,660,1000);
delay(1000);
}
```

Aufgaben

1. Schreibe das Programm ab und speichere den zugehörigen Sketch unter „05_Lautsprecher“.
2. Was wirst du hören, wenn du einen Lautsprecher an Port 11 anschließt und das Programm von **M 9** auf den Arduino überträgst? Überlege zuerst und überprüfe anschließend deine Vermutung.
3. Bestimme den Hörbereich deines Ohres und vergleiche ihn mit dem deines Nachbarn und von einzelnen Lebewesen in der Tabelle.
4. Was ist bei nebenstehendem Programm zu hören?
5. Programmiere eine Melodie deiner Wahl.
6. Ergänze deine Ampelkreuzung um die Ausgabe eines Tones bei der Fußgängerampel, wenn diese Grün anzeigt.
7. Speichere den zugehörigen Sketch unter „06_Ampelkreuzung_Ton“.
8. Ergänze dein Glossar um die *tone*- und *notone*-Befehle.

längeres Beinchen an PIN 7 und kürzeres Beinchen an GND



Glossar

Arduino-Befehl	Beschreibung
Void setup { }	Hier werden Grundeinstellungen (z. B. ob ein Kanal ein In- oder Output ist) vorgenommen. Diese Methode wird nur einmal beim Programmstart ausgeführt.
Void loop { }	Die Methode wird im Gegensatz zu „void setup" ständig wiederholt. Hier wird der eigentliche Programmablauf hineingeschrieben.
pinMode(3,OUTPUT);	Dieser Befehl setzt den digitalen PIN 3 als Ausgang fest.
pinMode(3,INPUT);	Dieser Befehl setzt den digitalen PIN 3 als Eingang fest.
digitalWrite(3 HIGH);	Dieser Befehl „schaltet" den digitalen PIN 3 an.
digitalWrite(3,LOW);	Dieser Befehl „schaltet" den digitalen PIN 3 aus.
delay(2000);	Dieser Befehl erreicht eine Pause in der Programmausführung in Höhe von 2000 Millisekunden, also 2 Sekunden.
int waiting = 5000; delay(waiting);	Diese Befehle definieren eine Variable namens *waiting* und eine Pause in Höhe von 5 Sekunden. Nur bei „int" muss man diese Zahl ändern.
tone(11, 440, 1000);	Port 11 wird 440 Mal pro Sekunde für 1000 Millisekunden (1 Sekunde) an- und ausgeschaltet.
notone(11);	Mit diesem Befehl schaltet man die Tonausgabe am Port 11 ab.
Serial.begin(9600);	Legt die Datenrate in Bit pro Sekunde (Baud) für die serielle Datenübertragung fest.
analogRead(LDR);	Dieser Befehl liest den Wert vom angegebenen analogen Pin, in diesem Fall dem LDR, ein.
Serial.println(Helligkeit);	Dieser Befehl druckt Daten an den seriellen Anschluss als von Menschen lesbarer ASCII-Text, gefolgt von einem Zeilenwechsel.
digitalRead(Taster);	Dieser Befehl liest den Wert vom angegebenen digitalen Pin, in diesem Fall des Tasters, ein.
if (tasterstatus == HIGH) *{* *…* *}* *else* *{* *…* *}*	Erster Teil: Bedingung (wenn / if): Zu Beginn wird geprüft, ob die Bedingung erfüllt ist. Zweiter Teil: Bedingung erfüllt (dann / then): Ist die Bedingung erfüllt, dann führt der Arduino das Programm in den ersten geschwungenen Klammern aus. Dritter Teil: Bedingung nicht erfüllt (ansonsten / else): Ist die Bedingung nicht erfüllt, dann führt der Arduino das Programm in den zweiten geschwungenen Klammern aus.

Hinweise (M 1 und M 2; 1./2. Stunde)

Ziel der ersten beiden Stunde ist es, die Schüler auf den Einsatz von Mikrocontrollern im Allgemeinen neugierig zu machen und in technische Aspekte einzuführen.
Im **Einstieg** zu der Stunde machen sich die Schüler darüber Gedanken, welche einzelnen **Funktionen ausgewählte technische Geräte** haben und was ihnen dabei gemeinsam zu sein scheint. Dazu legen Sie als Lehrkraft zunächst die Folie (**M 1)** auf und lassen in einem **Unterrichtsgespräch** die zugehörigen Aufgaben bearbeiten und verschriftlichen.
In der sich anschließenden **Erarbeitungsphase** geht es in einem **Unterrichtsgespräch** unter Einsatz des Arbeitsblattes **M 2** nun darum, **die technischen Grundlagen der Funktionsweise eines Mikrocontrollers** zunächst am Beispiel eines Autoschlüssels zu besprechen und sie in einem zweiten Schritt mit dem **Eingabe-Verarbeitung-Ausgabe-Prinzip** zu verallgemeinern. Sämtliche Ergebnisse werden anschließend gesammelt und stichwortartig festgehalten.
Den Abschluss der Unterrichtsstunde bildet **Organisatorisches**, um in der nächsten Stunde gleich mit dem Mikrocontroller Arduino starten zu können. Es empfiehlt sich, dass für die kommenden Unterrichtsstunden jeder Schüler seine einzelne Starter-Box hat, in der ein Zettel mit seinem Namen liegen sollte. Jeder Schüler sollte entweder einen USB-Stick in seinem Mäppchen regelmäßig dabeihaben oder einen solchen in der Box ablegen. Auf diesem können die erstellten Programme gespeichert und bearbeitet werden. Zur schnellen Wiedererkennung ist es hilfreich, einen Zettel mit Namen des Schülers und Klasse sichtbar in die Box zu legen, wenn diese Boxen in einem Raum separat gelagert werden. Das Word-Dokument „Arduino – Organisation" bietet eine mögliche Vorlage. Alternativ kann jeder Schüler natürlich seine Box in den entsprechenden Stunden auch immer wie einen Füller oder ein Geodreieck mitbringen.

Erwartungshorizont (M 1)

Aufgabe 1: Waschmaschine: Nach Einfüllen des Pulvers wird – abhängig von der Art der Wäsche – ein bestimmtes Programm durch Tastendruck ausgewählt. Ein Mikrocontroller verarbeitet diesen Befehl. In der Folge läuft dieses Programm mit einer entsprechenden Dauer, mit einer bestimmten Temperatur und ggf. mit weiteren Extras ab.
Kaffeemaschine: Es sind Wasser und Kaffeebohnen vorhanden bzw. nachzulegen. Man stellt seine Kaffeetasse drunter und wählt eine bestimmte Kaffeeart. Die Kaffeemaschine beginnt zu arbeiten und füllt zum Schluss die Kaffeetasse mit dem gewünschten Produkt.
Mikrocontroller: Auf den Chip eines Mikrocontrollers wird ein zuvor geschriebenes Programm geladen. Die Recheneinheit (CPU, Prozessor, Controller) greift darauf zu, speichert es zwischen und berechnet daraus die Datenausgabe. Anschließend werden bestimmte Befehle ausgeführt.
Tablet / Handy / Notebook: Auf diesen Geräten sind verschiedene Programme bzw. Apps installiert, die ausgewählt werden. In einem Programm bzw. einer App. selbst werden durch Finger- bzw. Tastendruck bestimmte Befehle angefordert, verarbeitet und anschließend umgesetzt.
Smartwatch: Auf solch einer Uhr sind verschiedene Funktionen vorhanden, die über eine Menüsteuerung oder per Tastendruck ausgewählt bzw. angefordert werden. Nach einer Verarbeitung wird der gewünschte Befehl ausgeführt.
Aufgabe 2: In allen Geräten müssen Steuerungen/Minicomputer/Mikrocontroller eingebaut sein.
Aufgabe 3: Individuelle Schülerantworten wie z. B.: eine sich selbsttätig öffnende Schiebetür: Eine Person nähert sich einer sich selbsttätig öffnenden Schiebetür. Tritt ein bestimmter Abstand ein, so öffnet sich die Tür. Ist diese Person durchgegangen bzw. keine weitere Person unterhalb eines bestimmten Abstandes, schließt sie sich wieder.