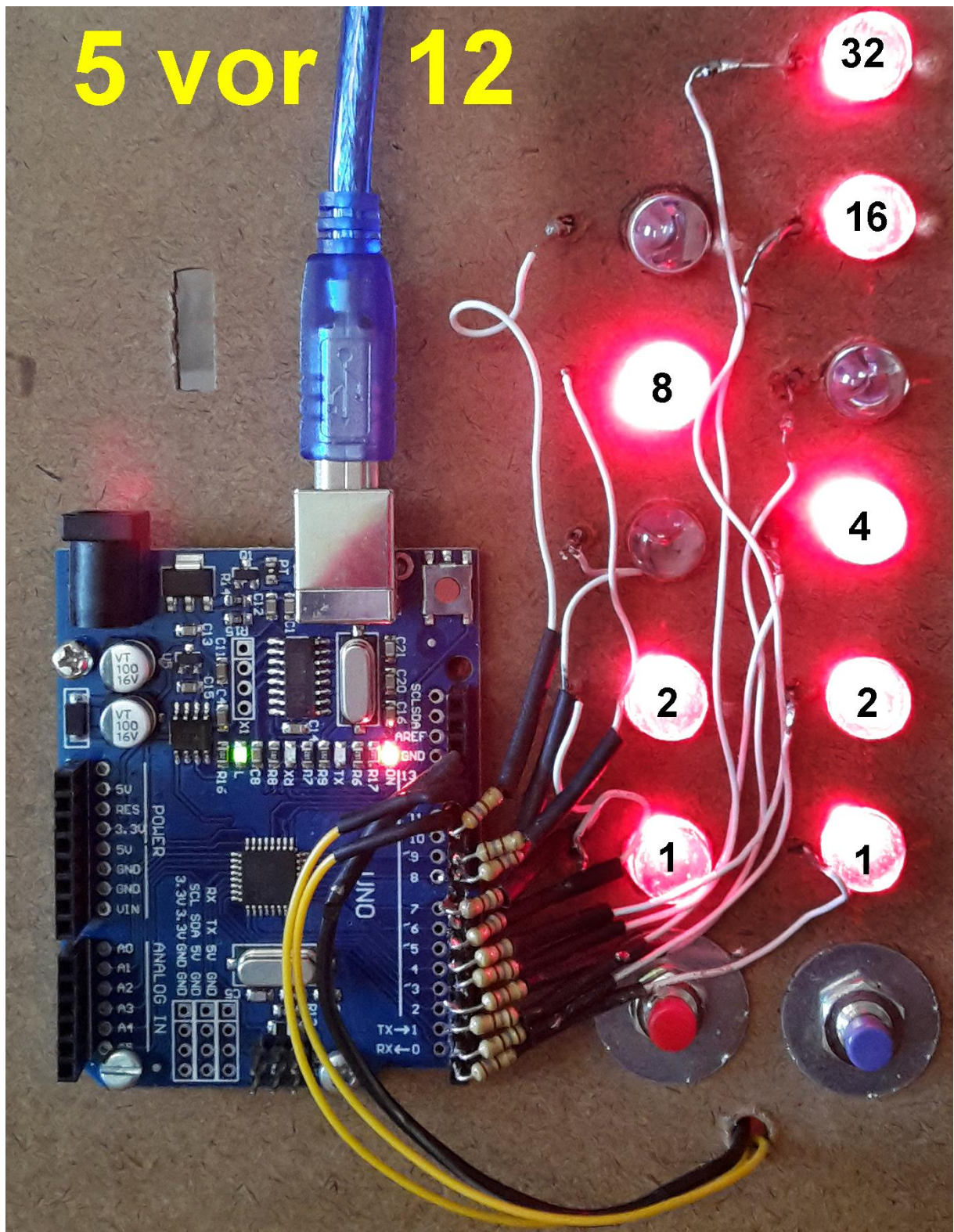


Binäruhr -



selbstgebaut mit Arduino

Fahrplan:

1. Elektrische Messgrößen und ihre Einheiten im Überblick
2. Binäruhr in Rohform beobachten und zu lesen versuchen
Arbeitsblatt "Binärzahl₂ = Dezimalzahl₁₀"
3. Komponenten identifizieren, Kosten durch Recherche ermitteln
4. Spannungsmessungen mit einem Digitalmultimeter
5. LED mit Widerstand an 4,5-Volt-Blockbatterie anschließen
6. LED mit Widerstand verlöten
7. Arduino Crashkurs: Digital OUT => selbstgelötete LED mit Widerstand blinkt,
Digital IN => ein Taster wird abgefragt
8. Digital in: Taster mit Pulldown-Widerstand
9. Gesamtschaltplan der Binäruhr
10. Designskizzen, Bau, Tests, Fehlersuche und –behebung, Präsentation ...
11. Möglicher Code für die Binäruhr

Aufgabe

Elektrische Messgrößen und ihre Einheiten im Überblick

Messgröße	Abkürzung für die Messgröße	Maßeinheit	Abkürzung für die Maßeinheit
<i>Ladung</i>	Q		
	I		
	U		
	R		
	P		

Schreibe ausführlich:

U = 4,5 V	
I = 20 mA	
R = 470 kΩ	
Q = 30 μC	<i>Die Ladung beträgt 30 Mikrocoulomb.</i>
P = 9 MW	

Schreibe abgekürzt:

Die Stromstärke beträgt 50 Mikroampere.	
Der Widerstand beträgt 1 Megaohm.	
Die Leistung beträgt 10 Watt.	
Die Ladung beträgt 5 Millicoulomb.	Q=5mC
Die Spannung beträgt 12 Volt.	

Binärzahl₂ = Dezimalzahl₁₀

$$1011_2 = 8_{10} + 0_{10} + 2_{10} + 1_{10} = 11_{10}$$

$$1010_2 = \quad \quad \quad = \quad 10$$

$$1111_2 = \quad \quad \quad = \quad 10$$

$$13_{10} = 1101_2$$

$$\begin{array}{l}
 13:2 = 6 \text{ R } 1 \\
 \rightarrow 6:2 = 3 \text{ R } 0 \\
 3:2 = 1 \text{ R } 1 \\
 1:2 = 0 \text{ R } 1
 \end{array}$$

$$\text{Probe: } 1101_2 = 8_{10} + 4_{10} + 0_{10} + 1_{10} = 13_{10}$$

$$5_{10} = 0 _ _ _ 2$$

$$\begin{array}{l}
 5:2 = \text{ R } \\
 2:2 = \text{ R } \\
 1:2 = \text{ R }
 \end{array}$$

Probe:

$$9_{10} = _ _ _ _$$

Probe:

Arbeitsblatt:

Komponenten identifizieren, Kosten recherchieren

Fachbegriffe und Handelsnamen:

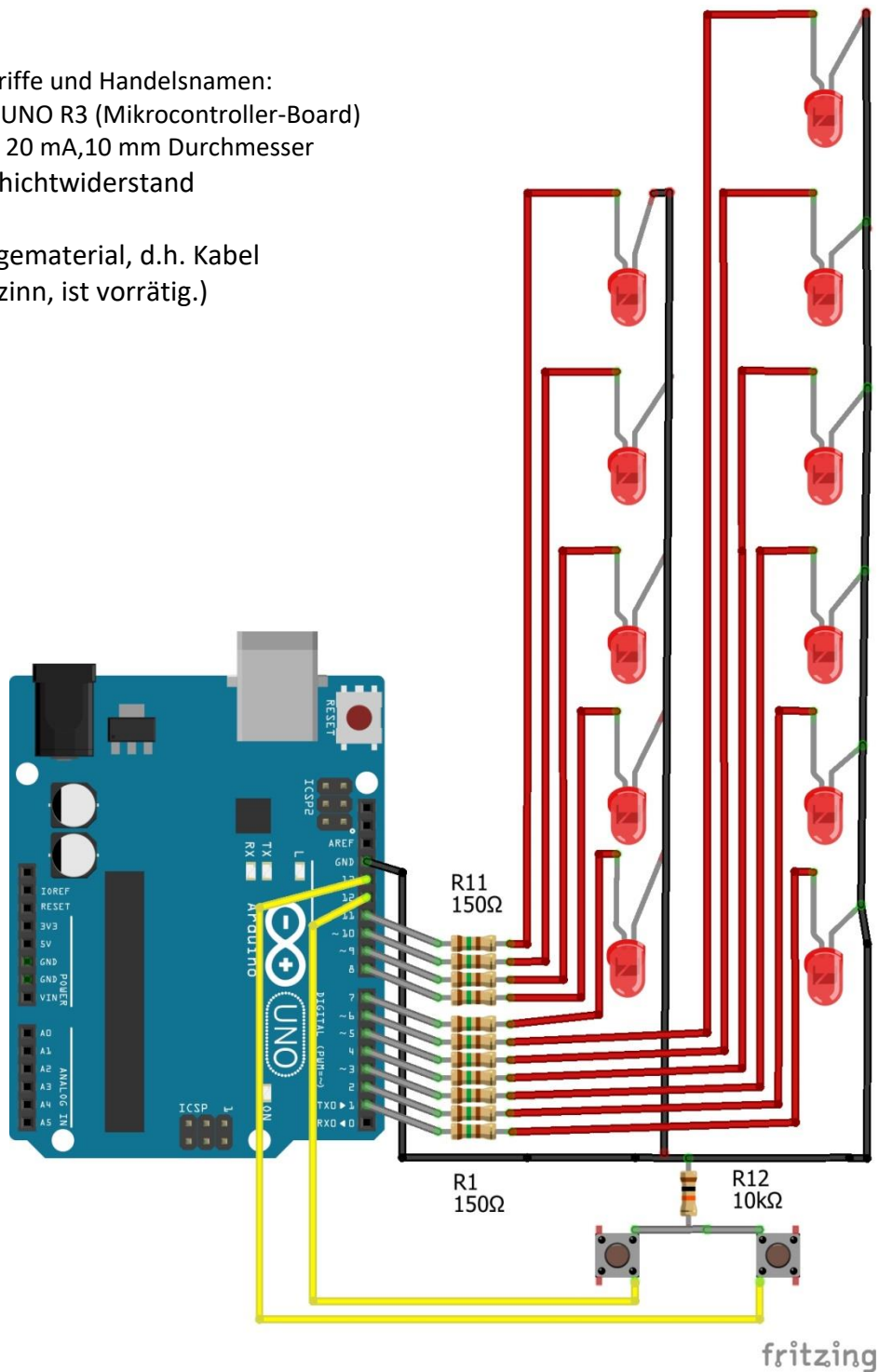
Arduino UNO R3 (Mikrocontroller-Board)

LED, rot, 20 mA, 10 mm Durchmesser

Kohleschichtwiderstand

Taster

(Montagematerial, d.h. Kabel
und Lötzinn, ist vorrätig.)



Hier die Rechercheergebnisse notieren:

Erklärungen und Regeln für das Digitalmultimeter (DMM)

Ein Digitalmultimeter, abgekürzt DMM, ist ein Messgerät, dass du für viele Experimente zu elektrischen und elektronischen Bauteilen gebrauchen kannst. Auch bei Reparaturen von elektrischen und elektronischen Geräten kann es nützlich sein. Das Wort "Multimeter" bedeutet das gleiche wie "Vielfachmessgerät", weil viele verschiedene elektrische Messgrößen mit einem Multimeter gemessen werden können: Spannungen, Stromstärken und Widerstände in den Einheiten mV (Millivolt), V (Volt), mA (Milliampere), A (Ampere), Ω (Ohm), k Ω (Kiloohm).

Spannungen und Stromstärken können als Gleichstrom (=) und teilweise als Wechselstrom (~) gemessen werden.

(Beim = Zeichen ist der untere Strich u.U. gestrichelt. Das bedeutet, es werden auch zerhackte Gleichströme gemessen).

Weil es außer den digitalen Messgeräten auch analoge gibt, sagt man das "Digital" bzw. "D" dazu. Analoge Messgeräte zeigen die Messwerte mit einem Zeiger und einer Skala an, digitale Messgeräte geben die Messwerte gleich als Zahl an. Früher gab es nur analoge Vielfachmessgeräte, heute sind die digitalen Multimeter billiger und weiter verbreitet. Wenn man einige der hier genannten Bezeichnungen und Abkürzungen noch nicht kennt, muss man in Physikbüchern, Elektronikbüchern bzw. im Internet recherchieren, was sie bedeuten oder Mitschülerinnen, Mitschüler, Lehrerinnen, Lehrer usw. fragen.

Auf jeden Fall sollte man auswendig lernen, welche Maßeinheiten und Abkürzungen zu welchen Messgrößen gehören! Folgende Gefahrenregeln darf man beim Umgang mit einem Multimeter niemals außer Acht lassen, da sonst Lebensgefahr bestehen kann:

- **Niemals Messungen an Stromkreisen mit mehr als 20 Volt Spannung durchführen!**
- **Niemals Messungen an Geräten, die an eine Steckdose angeschlossen sind, es sei denn, die Lehrerin oder der Lehrer gibt ausdrückliche Anweisungen!**
- **Niemals Geräte, die an eine Steckdose angeschlossen sind, öffnen. Vorher immer den Netzstecker ziehen!**
- **Auf gar keinen Fall Messungen direkt an Steckdosen (220 Volt, Starkstrom) durchführen.**

Damit das Digitalmultimeter nicht kaputt geht, müssen auch folgende Regeln beachtet werden:

1. Das blaue oder schwarze Messkabel kommt immer in die "COM" Buchse des Multimeters. (COM ist die Abkürzung für das englische Wort "common" und bedeutet „gemeinsam“. Es ist damit ein gemeinsamer Messpunkt gemeint - in Deutschland sagt man auch Erdung dazu - und das bedeutet, dass er meistens - nicht immer – auf einer Spannung von 0 Volt liegt.)
2. Überlege, was gemessen werden soll, und stecke das rote Messkabel dann in die passende Buchse des Multimeters. Es ist fast immer die mittlere Buchse, an der A, V und Ω stehen. Nur dann, wenn man Stromstärken größer als 0,2 Ampere (das sind 200 Milliampere) messen will, dann muss man die 10-Ampere-Buchse wählen, die mit 10 A beschriftet ist.
3. Stelle den richtigen Messbereich ein, BEVOR du die Messkabel mit der Schaltung in Kontakt bringst, an der gemessen werden soll. Entscheide vorher, ob du Stromstärken (A, mA), Spannungen (V, mV) als Gleich- oder Wechselstrom messen willst. Wähle dann einen Messbereich, der größer als der größte Messwert ist, den du erwartest. Bei einer 1,5-Volt-Batterie ist es z.B. 2-Volt-Messbereich, bei 4,5-Volt oder 9 Volt-Batterien der 20-Volt-Messbereich. Bei Stromstärken muss man oft rechnen, was der richtige Messbereich ist.

Wenn man bei Spannungs- und Stromstärkemessungen nicht vorsichtig ist, fließt unter Umständen ein zu hoher Messstrom durch das Messgerät und beschädigt dieses. Falls man während der Messung merkt, dass man sicher unter dem Wert des nächst kleineren Messbereichs liegt, dann kann man auf ihn herunterschalten, damit die Messung genauer wird. Bei Widerstandsmessungen (Ω , k Ω) muss man nicht so vorsichtig sein, weil das Messgerät selbst den Messstrom liefert. Man darf aber niemals an Widerständen messen, die in einer Schaltung eingebaut und/oder an eine Stromquelle angeschlossen sind. Die Schaltung und das Messgerät könnten zerstört werden. Außerdem ist das Messergebnis – falls es überhaupt zu Stande kommt – nicht der Widerstandswert des Bauteils, sondern der gesamten Schaltung.

Fragen zum DMM

1. Was bedeutet die Abkürzung „DMM“?
2. Welche elektrischen Messgrößen lassen sich mit einem DMM erfassen und welche Einheiten gehören jeweils dazu?

Messgröße	Abkürzung für die Messgröße	Maßeinheiten	Abkürzungen für die Maßeinheiten
	U		
			A
		Ohm, Kiloohm	

3. Welche Sicherheitsregeln für Messungen mit einem Vielfachmessgerät müssen im Schulunterricht beachtet werden? Fasse stichwortartig zusammen!

4. Welche Messungen können das DMM zerstören, wenn die Messbereiche vorher nicht sorgfältig gewählt werden, und warum ist das so?

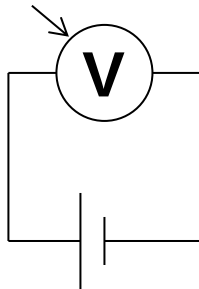
5. Die tatsächliche Spannung einer 6-Volt-KFZ-Batterie (Nennspannung) in einem Motorroller soll gemessen werden. Welcher Messbereich muss eingestellt werden?



Spannungen an einer Flachbatterie (unbelastet/belastet)

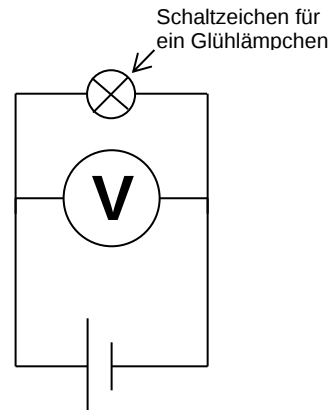
1. Miss die Leerlaufspannung einer Flachbatterie.
2. SchlieÙe ein Glühlämpchen an und miss die Spannung der Batterie.
3. SchlieÙe zwei Glühlämpchen in Reihenschaltung an und miss die Spannung.
4. SchlieÙe zwei Glühlämpchen in Parallelschaltung an und miss die Spannung.

Schaltzeichen für ein Spannungsmessgerät, auch Voltmeter genannt. Ein entsprechend eingestelltes Multimeter wird auch so gezeichnet.



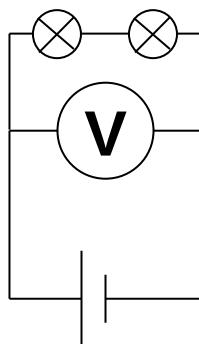
Leerlauf:

U = _____



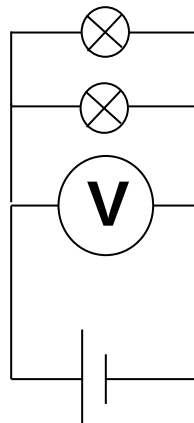
Ein Glühlämpchen:

U = _____



Zwei Glühlämpchen in Reihenschaltung:

U = _____



Zwei Glühlämpchen in Parallelschaltung:

U: _____

Auswertung:

- a) Ob mehr oder weniger Strom fließt, erkennen wir daran, _____.
- b) Kein Strom fließt in der Schaltung mit der Bezeichnung _____.
- c) Besonders viel Strom fließt _____.
- d) Wenig Strom fließt _____.
- e) Wenn viel Strom fließt, dann ist die Spannung _____.

Wie schließt man eine Leuchtdiode richtig an?

Die Leuchtdiode (LED) erkennst du an dem halbrunden Kopf (siehe Bild unten), welcher unterschiedlich eingefärbt oder klar sein kann. Nimm die LED und schau sie dir aus der Nähe an. Du wirst sie heute noch zum Leuchten bringen. Aber aufgepasst!

Achtung:

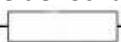
Eine Leuchtdiode darf nicht direkt an eine 4,5-Volt-Batterie angeschlossen werden. Es würde ein zu starker Strom fließen und die LED würde zerstört werden.

Um eine Leuchtdiode richtig an eine Batterie anzuschließen, brauchst du ein Bauteil, das Widerstand heißt. Widerstände werden mit Farbringen gekennzeichnet. Für den Anschluss einer Leuchtdiode an eine 4,5-Volt-Stromquelle braucht man einen Widerstand mit 150 bis 220 Ohm. Seinen Wert kannst du mit einer Farbcodetabelle überprüfen. Sicherer ist es aber, du misst mit einem Messgerät nach.

Aufgabe 1: Schraube die unten aufgezeichnete Schaltung mit Hilfe von Lüsterklemme und einem passenden Kabelstück zusammen. Baue die Schaltung so auf, dass die Leuchtdiode leuchtet, wenn der Widerstand an den Pluspol und das blaue Ausgleichskabel an den Minuspol der Batterie gehalten wird. Beachte, dass die Leuchtdiode den Strom nur in eine Richtung hindurchlässt.

Aufgabe 2: Benenne die Abbildung mit Hilfe der unterstrichenen Wörter!



Aufgabe 2: Zeichne mit Hilfe der Schaltzeichen für eine Leuchtdiode  und für einen Widerstand  den Schaltplan für die aufgebaute Schaltung. Die Zeichen für Batterie und Kabel solltest du kennen, für die Lüsterklemme gibt es kein Schaltzeichen, man zeichnet die Kabel bzw. Drähte einfach weiter.

Gefahren beim Löten



Verbrennungen durch den heißen LötKolben



Verbrennungen durch Tropfen von heißem Lötzinn



Verbrennungen durch heiße Bauteile



Beschädigung von Kleidung und anderen Gegenständen durch diese Gefahrenquellen

Richtig löten



Arbeitsplatz aufgeräumt halten!
(Kein überflüssiges Papier etc.)



Material und Werkzeug bereitlegen!



Erst die Bauteile erhitzen, dann das Lötzinn zuführen!



Lötzinn genau dosieren!

Beim Löten immer auf gute Belüftung achten – keinen bleihaltiges Lötzinn verwenden!

Lötaufgabe: Verlöte die LED, den Widerstand, das Kabelstück und den kleinen Pfostenstecker zu einer Schaltung. Diese soll sich ohne weiteres Montagematerial an einen Arduino anschließen lassen – siehe auch nächste Seite!

Leuchtdiode

Schrumpfschlauch

Widerstand

Kabelstück

Pfostenstecker

Wichtige Hinweise:

Für das Löten ist Partnerarbeit ideal!

Vor dem Löten muss die Isolierung von den Kabelstellen entfernt werden, die verlötet werden sollen. Man sagt, die Kabel werden „abisoliert“. Dazu ist Übung erforderlich.

Werkzeug: Abisolierzange, Saitenschneider oder Schere.

Beim Löten müssen Kabel und Bauteile zunächst verzinnt werden. Dazu müssen sie erhitzt werden, bevor das Lötzinn aufgeschmolzen wird.

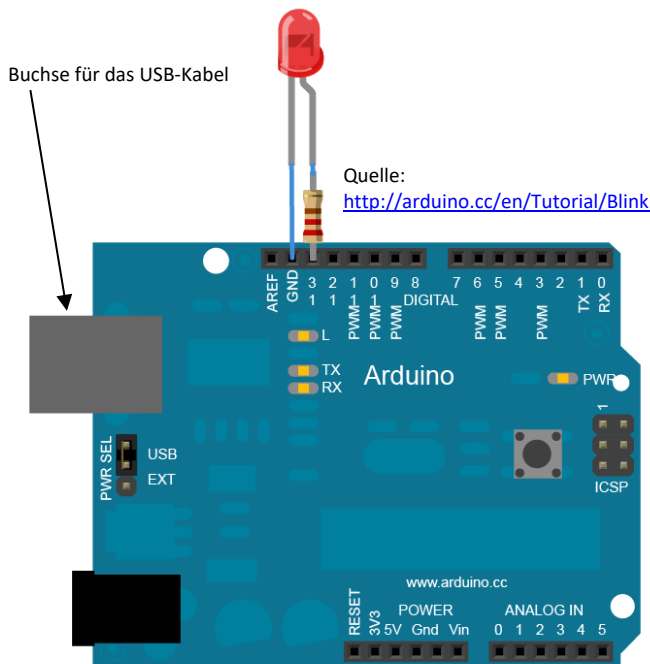
„Drauftropfen lassen“ reicht nicht, das gibt „kalte“ Lötstellen, die häufig zu Kontaktproblemen führen.

Stellen, bei denen es zu Kurzschlüssen kommen kann, müssen isoliert werden. Besonders gut geeignet ist dazu sogenannter „Schrumpfschlauch“. Der zieht sich bei Erhitzung zusammen und fixiert sich so an der kritischen Stelle.

Crashkurs: Arbeiten mit einem Arduino-Mikroprozessor-Board

An dieser Station beschränken wir uns darauf, die Mikroprozessor-Programmierung und seine Arbeitsweise an einem möglichst einfachen Beispiel zu zeigen. Die Informationen sind auf folgender Internetseite zu finden: <http://arduino.cc/en/Tutorial/Blink>. Von <http://arduino.cc> kann man auch die Software zur Arduino-Programmierung (Entwicklungsumgebung) herunterladen.

Die notwendigen Informationen zur Arduino-Programmierung sind zu umfangreich, um sie hier sinnvoll darstellen zu können und man findet sie gut aufbereitet im Internet.



Notwendige Hardware:

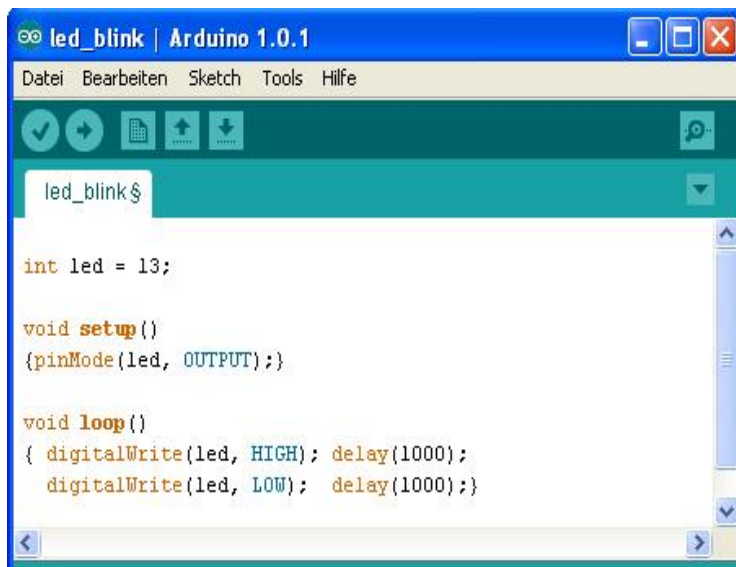
- Computer
- USB-Kabel
- Arduino Board
- LED
- Widerstand 220 Ohm

Für diese Station wurde eine LED passend zur Abbildung mit dem Widerstand verlötet (s.o.).

Aufbau:

Die LED wird - wie im Bild gezeigt – an Pin 13 und GND angeschlossen. Dann wird der Arduino mit dem USB-Kabel mit dem Computer verbunden, damit mit dem Programm „arduino.exe“ der Sketch „led_blink.ino“ erstellt und auf den Mikroprozessor hochgeladen werden kann. Der Begriff „Sketch“ hat eine entsprechend Bedeutung wie „Programm“ oder „App“.

(Der Name des Sketches, hier „led_blink.ino“, wird erst beim Abspeichern vergeben.)



Der „Witz“ an der ganzen Sache mit dem Mikroprozessor – egal ob Arduino oder anderer Bauart – ist nun: Auch wenn das Programmierkabel entfernt wird und die Stromversorgung (hier über Pin „Vin“) anders sichergestellt ist, führt er seine Funktion, z.B. das Blinken, ebenfalls aus!

Im Programmierfenster muss nur das fett Gedruckte stehen, siehe links:

int led = 13;

Der Integer-Variablen „led“ wird die Zahl 13 für Pin 13 zugewiesen.

void setup()

{pinMode(led, OUTPUT);}

Im Programmteil (der Funktion) setup(){...}, der zu jedem Sketch gehört, werden Einstellungen vorgenommen, hier: Pin „led“ – also 13 – erhält die Funktion „OUTPUT“.

void loop()

**{ digitalWrite(led, HIGH); delay(1000);
digitalWrite(led, LOW); delay(1000);}**

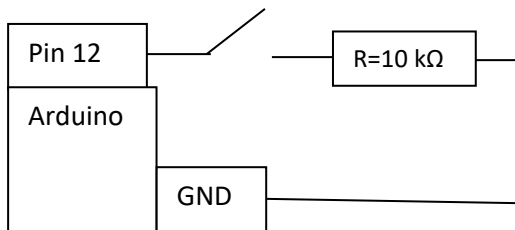
In der Funktion loop() {...}, sozusagen dem Hauptprogramm, die ebenfalls zu jedem Sketch gehört, wird festgelegt, was der Arduino immer wieder machen soll – solange der Strom eingeschaltet bleibt. 1000 Millisekunden „HIGH“, d.h. 5 Volt an Pin 13, dann 1000 ms „LOW“, d.h. 0 V.) (Das Programm nimmt die farbigen Markierungen selbst vor.)

Digital in: Taster mit Pulldown-Widerstand

Um in das laufende Programm eine Eingabe zu machen, kann man einen einfachen Tastschalter verwenden. Der darf allerdings nicht ohne Absicherung an einen Pin angeschlossen werden, das könnte zu Kurzschlüssen oder Funktionsstörungen führen.

Die saubere Lösung sieht folgendermaßen aus: Mit dem Befehl „pinMode (12, INPUT)“ wird Pin 12 als Eingabepin vorbereitet, mit „digitalWrite(12,HIGH)“ wird anschließend 5 Volt Spannung vom Arduino an den Pin gelegt.

Nun kann mit dem Befehl „testSpannung=digitalRead(12)“ der Zustand von Pin 12 abgefragt und der Variablen „testSpannung“ zugewiesen werden. Weil die Spannung 5 Volt beträgt, hat „testSpannung“ im geschilderten Fall den Wert „HIGH“.



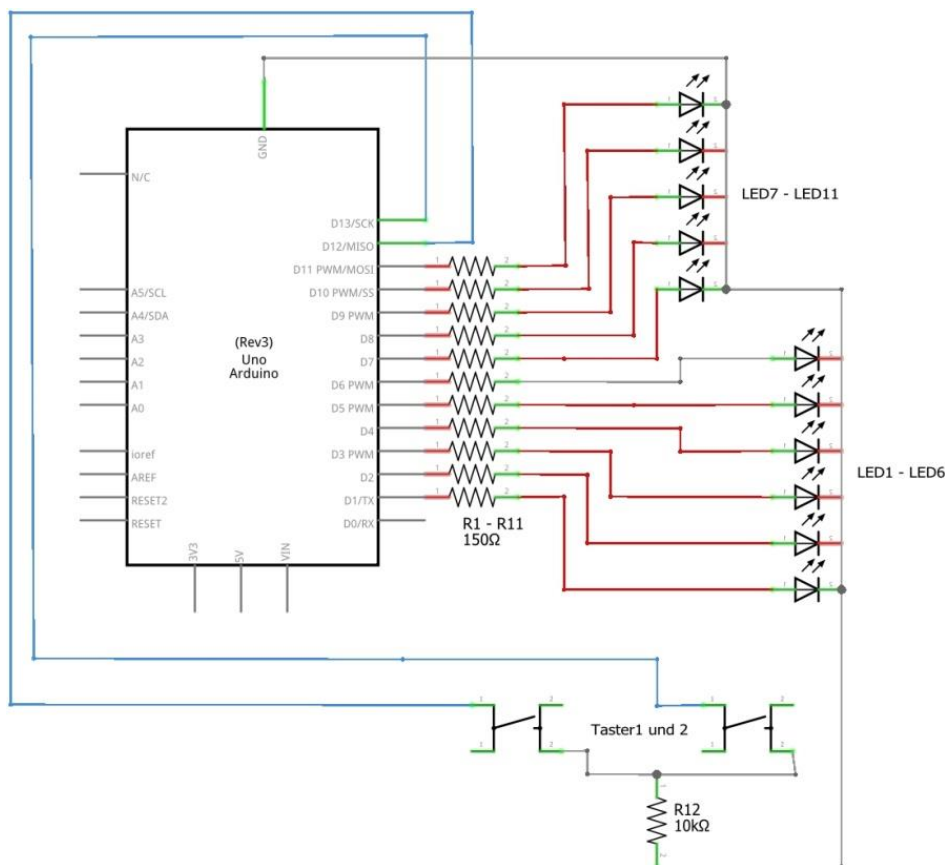
Nun wird an Pin 12 ein Taster angeschlossen, an dessen zweitem Anschluß ein 10-Kilohm-Widerstand $R=10k\Omega$ angebracht ist, der wiederum auf 0 Volt, also GND (Ground) – deutsch auch „Masse“ genannt – führt. Damit lässt sich Folgendes bewirken:
Wird der Taster gedrückt, fließt Strom aus Pin 12 durch den Taster und den Widerstand nach Masse. Dadurch sinkt die Spannung an Pin 12 auf fast 0 Volt.

Wird nun der Befehl „testSpannung=digitalRead(12)“ ausgeführt, erhält „testSpannung“ den Wert „LOW“. Das kann nun vom Programm ausgewertet werden.

Aufgabe: Verlöte einen Taster mit einem Pfostenstecker, einem Kabelstück und einem Widerstand wie in der Skizze oben angedeutet. Schreibe einen Sketch für eine Pinabfrage, lade ihn auf den Arduino hoch, teste Hard- und Software! Nimm dazu ein DMM und miss die Spannungswerte an Pin 12 bei gedrücktem und bei nicht gedrücktem Schalter. (Alternative mit erhöhtem Programmieraufwand: Der Arduino kann den Wert der Variablen „testSpannung“ ja nicht direkt anzeigen. Du kannst diesen Wert aber mit der seriellen Schnittstelle des Arduinos über das USB-Kabel auf den Rechner übertragen. Dort kann vom Monitorprogramm, das in der Arduino-Entwicklungsumgebung (IDE) integriert ist, der Wert angezeigt werden. Anleitungen dazu findet man im Netz.)

Notiere die Zeilen des Sketches hier:

Gesamtschaltplan der Binäruhr



Platz für Designskizzen, Berichte über Bau, Fehlersuche und Präsentationen

(ggf. Fotos einkleben, zusätzliche Blätter einlegen ...)

Möglicher Code für die Binäruhr

```
BinaeruhrLuGaKompaktcode | Arduino 1.8.2
Datei Bearbeiten Sketch Werkzeuge Hilfe

BinaeruhrLuGaKompaktcode $

int sekunde=0, minute=0, stunde=0,minutenuebertrag=0,stundenuebertrag=0,stellminute=0,stellstunde=0;
static unsigned long tick = 0;
void setup() {
  pinMode(1, OUTPUT);pinMode(2, OUTPUT);pinMode(3, OUTPUT);pinMode(4, OUTPUT);pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);pinMode(7, OUTPUT);pinMode(8, OUTPUT);pinMode(9, OUTPUT);pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);pinMode(12, INPUT);pinMode(13, INPUT);
  digitalWrite(12, HIGH);digitalWrite(13, HIGH);}
void loop() {
  if (millis() - tick >=1000) {tick = millis();sekunde++;}
  if (sekunde >= 60) {minute++;sekunde = 0;}
  if (minute >=60) {stunde++;minute = 0;}
  if (stunde >=24) {stunde=0;minute = 0;}
  if(minute %2 ==1) { digitalWrite(1, HIGH);} else { digitalWrite(1, LOW);}
  minutenuebertrag = minute/2;
  if(minutenuebertrag %2 ==1) { digitalWrite(2, HIGH);} else { digitalWrite(2, LOW);}
  minutenuebertrag = minutenuebertrag/2;
  if(minutenuebertrag %2 ==1) { digitalWrite(3, HIGH);} else { digitalWrite(3, LOW);}
  minutenuebertrag = minutenuebertrag/2;
  if(minutenuebertrag %2 ==1) { digitalWrite(4, HIGH);} else { digitalWrite(4, LOW);}
  minutenuebertrag = minutenuebertrag/2;
  if(minutenuebertrag %2 ==1) { digitalWrite(5, HIGH);} else { digitalWrite(5, LOW);}
  minutenuebertrag = minutenuebertrag/2;
  if(minutenuebertrag %2 ==1) { digitalWrite(6, HIGH);} else { digitalWrite(6, LOW);}
  if(stunde %2 ==1) { digitalWrite(7, HIGH);} else { digitalWrite(7, LOW);}
  stundenuebertrag = stunde/2;
  if(stundenuebertrag %2 ==1) { digitalWrite(8, HIGH);} else { digitalWrite(8, LOW);}
  stundenuebertrag = stundenuebertrag/2;
  if(stundenuebertrag %2 ==1) { digitalWrite(9, HIGH);} else { digitalWrite(9, LOW);}
  stundenuebertrag = stundenuebertrag/2;
  if(stundenuebertrag %2 ==1) { digitalWrite(10, HIGH);} else { digitalWrite(10, LOW);}
  stundenuebertrag = stundenuebertrag/2;
  if(stundenuebertrag %2 ==1) { digitalWrite(11, HIGH);} else { digitalWrite(11, LOW);}
  stellminute = digitalRead(12);
  if(stellminute== LOW) {minute++;sekunde=0;delay(300);}
  stellstunde = digitalRead(13);
  if(stellstunde==LOW) {stunde++;sekunde=0;delay(300);}
}
```

Bei „tick >= 1000“ sollte für erste Tests auf 10 gegangen werden, damit die Uhr 100mal schneller läuft und das grundsätzliche Funktionieren kontrolliert werden kann.

Erläuterungsbedürftig dürften der Variablentyp „unsigned static long“ und die Funktion „millis()“ sein:

Arduino Reference

https://www.arduino.cc/reference/en/language/functions/time/millis/

HOME BUY SOFTWARE PRODUCTS EDUCATION RESOURCES COMMUNITY HELP

LANGUAGE

FUNCTIONS

VARIABLES

STRUCTURE

LIBRARIES

GLOSSARY

The Arduino Reference text is licensed under a Creative Commons Attribution-Share Alike 3.0 License.

Find anything that can be improved? Suggest corrections and new documentation via GitHub.

Doubts on how to use Github? Learn everything you need to know in this tutorial.

Last Update: 11/15/2017

EDIT THIS PAGE

Reference > Language > Functions > Time > Millis

millis()

[Time]

Description

Returns the number of milliseconds since the Arduino board began running the current program. This number will overflow (go back to zero), after approximately 50 days.

Syntax

time = millis()

Parameters

Nothing

Returns

Number of milliseconds since the program started (unsigned long)