Co-funded by the Erasmus+ Programme of the European Union

‘We are the makers - IoT’ Learn-Scenario: Biofeedback with IoT-Biosensoren

Author: Thomas Jörg, Johannes-Kepler-Gymnasium Weil der Stadt

Das folgende Papier wurde im Schuljahr 2018/2019 in einem Schulumfeld mit ca. 18 Schülern im Alter von 13-17 Jahren entwickelt und getestet. Es spiegelt die Erfahrung mit vielen Verzweigungen und Misserfolgen wider. Da das IoT-Feld komplex ist, müssen Lehrmaterialien sorgfältig ausgewählt werden. Dieses Papier soll der Ausgangspunkt einer Empfehlung sein.

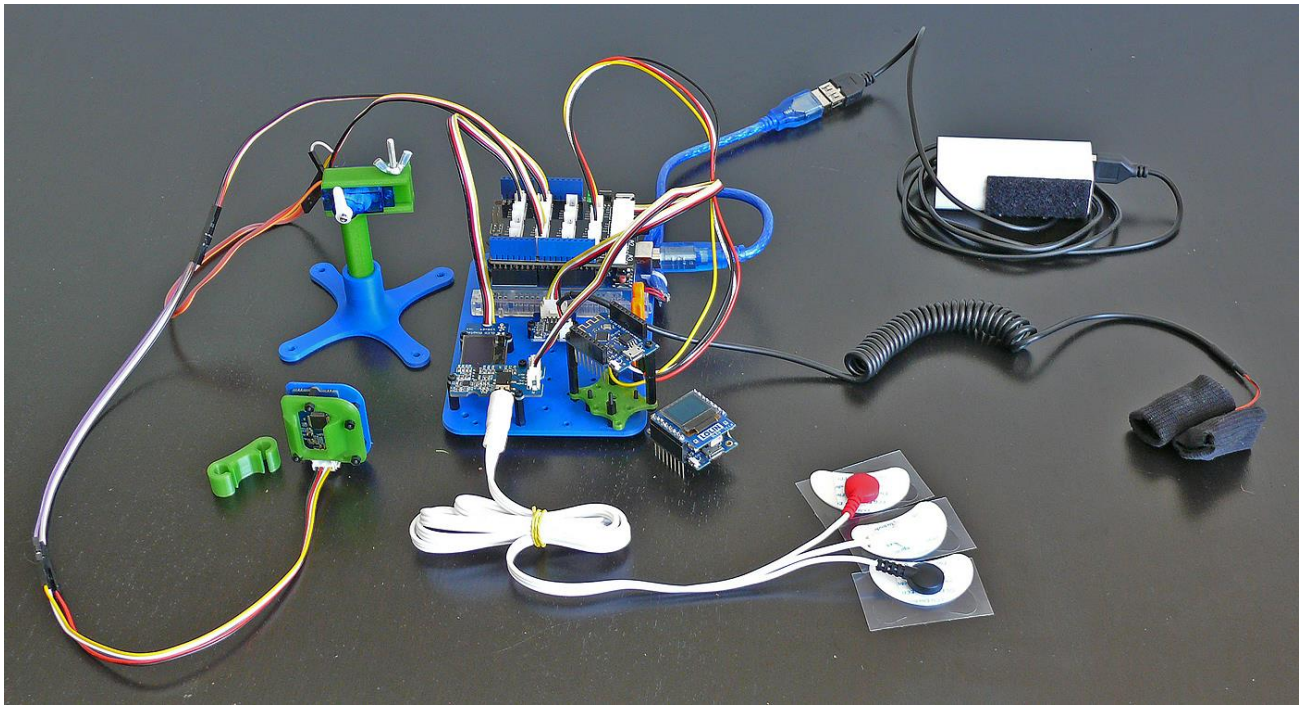
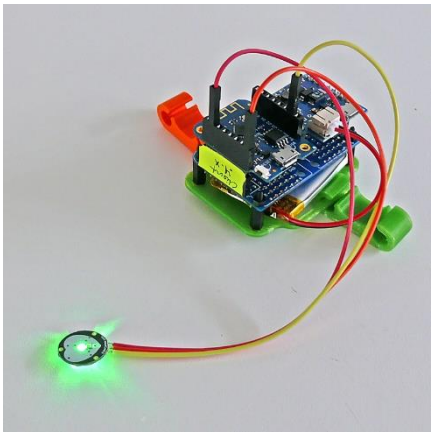


Abbildung 1Prototyp einer IoT-Biofeedback-Station



Was ist mit einem selbstgebauten Lügendetektor oder einem Fitness-Tracker mit selbstprogrammierter Steuerungssoftware? Oder einem Pulsometer, das mir hilft, mich zu entspannen und meine Belastung verfolgt?

Was ist mit einem Gerät, das mir Feedback zu meinem Muskeltonus gibt und mir hilft, mich in Stresssituationen zu besser zu kontrollieren? Die Steuerung einer Maschine mittels der Kraft meiner Gedanken – möglich oder nicht?

Und: Was ist mit Schlaf-Tracking-Geräten, die helfen, meinen Schlaf zu optimieren oder einen Arzt zu informieren, wenn ich nicht in Ordnung bin? Wo sind Möglichkeiten und Bedrohungen moderner IoT-Biosensor-Technologie?

Number of project: 2017-1-DE03-KA201-035615

1. Titel des Szenarios	**Biofeedback mit IoT-Biosensoren**
2. Zielgruppe	14 - 17 Jahre
3. Dauer	Mindestens 5 Wochen 2 *45min-Unterricht pro Woche: in Summe ca. 6-8 Stunden.
4. Lern-bedürfnisse, die durch die Übung abgedeckt werden	• Interaktion zwischen elektronischen Teilen und menschlichen Körpern • Überwachung und Beeinflussung menschlicher biologischer Parameter • Kommunikationskette von IoT-Geräten • Grundsätze von Sensoren und Aktoren • Verschiedene Prinzipien der Messung von Biosignalen. • EMG I: Wie funktioniert das Muskelleitungssystem? • EMG II: Prinzipien von Instrumentierungsverstärkern • Grundsätze drahtloser Kommunikationsnetze • Konstruktion und 3D-Druck von Messhelfern.
5. Erwartete Lernergebnisse	• Wie funktioniert ein IoT-System? • Wo liegen Möglichkeiten und Grenzen gesundheitsbasierter IoT-Systeme? • Welche Komponenten sind der Schlüssel zum Bau eines IoT-Geräts? • Wie kann man Biofeedback einsetzen, um Menschen zu helfen?
6. Methoden	In diesem Szenario konstruieren, bauen und programmieren die Kursteilnehmer selbst ein interaktives Biosignalgerät. Die Kursteilnehmer werden auch den Arduino Serial Monitor und Serial Plotter verwenden, um Biofeedback zu visualisieren und zu zeichnen.
7. Ort/Umgebung	• Ein Labor mit einer Reihe von elektronischen Teilen und Komponenten; • Jede Gruppe von Kursteilnehmern benötigt einen Computer oder Laptop mit Administratorrechten für die Installation verschiedener Softwarepakete • Ein Projektor für den Unterricht von Tutorials und die Präsentation von Studentenarbeiten; • jeder Schüler muss ein Labortagebuch führen

8. Werkzeuge/ Materialien/ Ressourcen

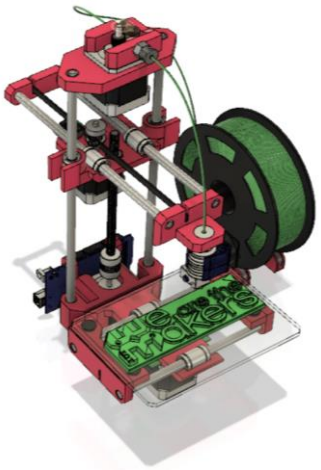


Abbildung 2 :Symbol des 3D-Druckers

3D-Drucker

Etwa 2-3 3D-Drucker sind notwendig, da die IoT-Biofeedback-Stationen ausgedruckt werden müssen. Natürlich ist es den Schülern möglich, Maschinenteile selbst zu konstruieren

3D-gedruckte Bauteile:

Als Ausgangspunkt werden alle notwendigen Teile im .stl-Format und als Autodesk Fusion 360-Dateien bereitgestellt.

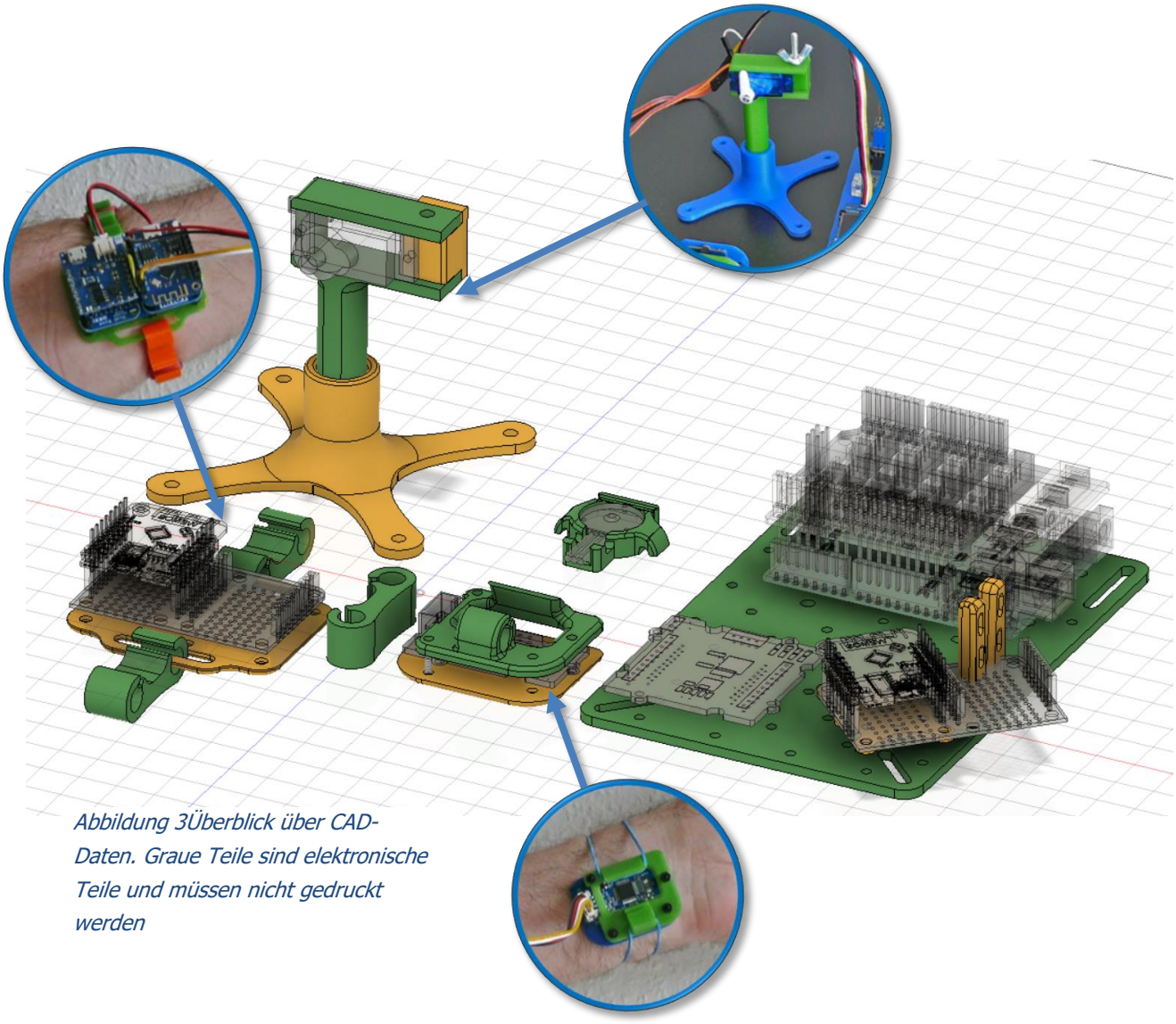


Abbildung 3Überblick über CAD-Daten. Graue Teile sind elektronische Teile und müssen nicht gedruckt werden

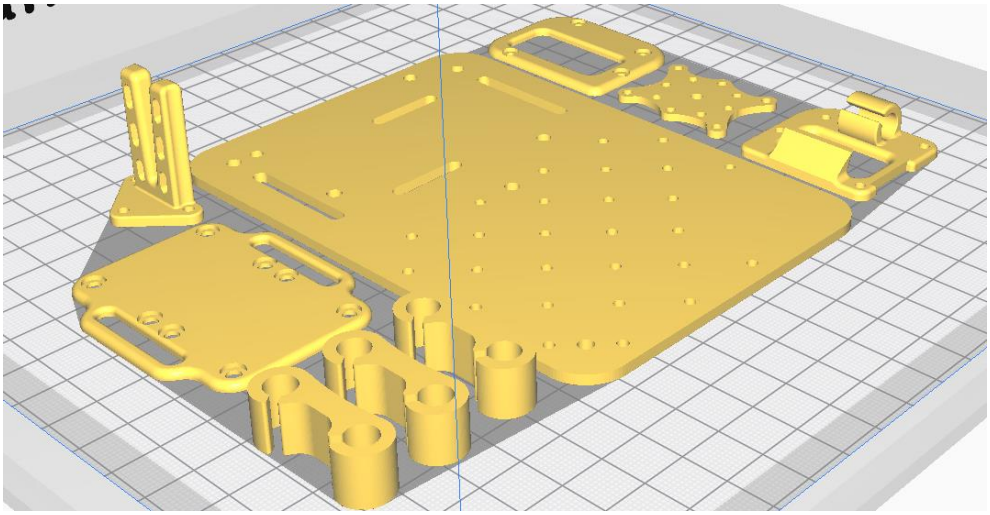
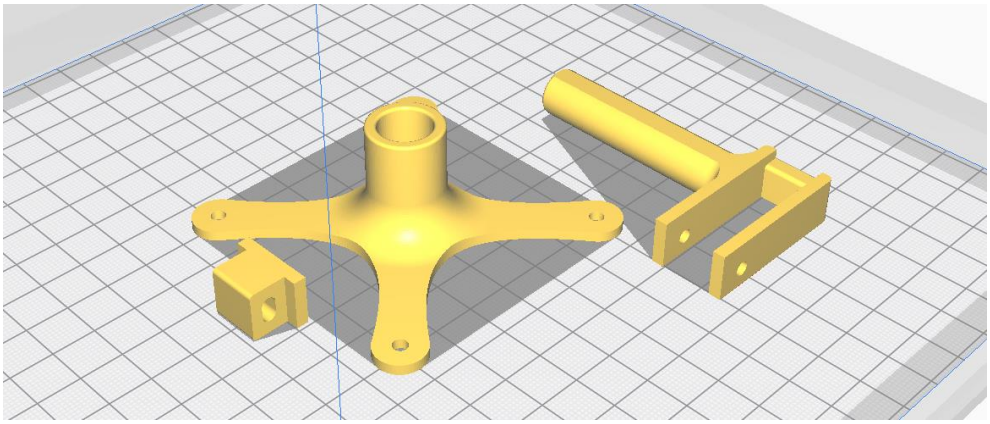


Abbildung 4 Satz von .stl-Dateien, druckfertig



Elektronische Komponenten:

Achtung: Da wir Experimente mit dem menschlichen Körper machen, muss jede Vorsichtsmaßnahme getroffen werden! Verbinden Sie niemals einen menschlichen Körper mit dem häuslichen Energiesystem. Der menschliche Körper muss immer komplett vom Stromnetz ferngehalten werden!

Dazu gehören auch Netzadapter, die an die Steckdose angeschlossen sind. Diese Art von Schaltungen muss vermieden werden. Verwenden Sie nur Batterien und Akkumulatoren mit einer Niedrigen Spannung von ca. 3-5V.

In dieser Arbeit empfehlen wir das Seeed Grove System als Basis für seine einfache Nutzung: (http://wiki.seeedstudio.com/Grove_System/) Alle Kernkomponenten außer Wemos Chips, Akkumulatoren und Herzfrequenzsensoren gehören zum Grove Standard:

Seeed Studio Komponenten:

1: Grove Base Shield für Arduino-Uno (http://wiki.seeedstudio.com/Base_Shield_V2/)

2. Grove OLED 128x64 (http://wiki.seeedstudio.com/Grove-OLED_Display_0.96inch/)

3. Grove EMG Detector (http://wiki.seeedstudio.com/Grove-EMG_Detector/)

4. Grove Fingerclip Herzfreq. http://wiki.seeedstudio.com/Grove-Finger-clip_Heart_Rate_Sensor/

5. Grove GSR http://wiki.seeedstudio.com/Grove-GSR_Sensor/

Normale Sensoren and Aktoren:

1: Arduino Uno (or gleichwertig)

A: **2x** Wemos LOLIN D1 mini (or gleichwertig)

B: Analoger Pulssensor für den Finger (www.pulsesensor.com)

C: Wemos battery shield (https://wiki.wemos.cc/products:d1_mini_shields:battery_shield)

D: Micro Servo Motor.

Sonstige Teile:

- Wundpflaster (um Sensoren an der Haut zu befestigen)
- Selbstklebendes Kupferband
- M3 Nylon Standoffs (Hex spacer)
- M2 Nylon Standoffs (Hex spacer) für Grove (has 2mm holes)
- Grove Kabel
- WAGO Klemmen
- Jumper wires
- USB Ladegerät das USB power pack
- USB power pack
- Lötkolben

Figure 5: Nylon Standoffs

Computer mit der folgenden Software vorinstalliert:

- Autodesk Fusion 360 (oder jede andere 3D-Modellierungssoftware, z.B. Wings3D)

CURA slicing software,

Eine Internetverbindung zum Herunterladen von Bibliotheken

- Arduino IDE
- Processing IDE

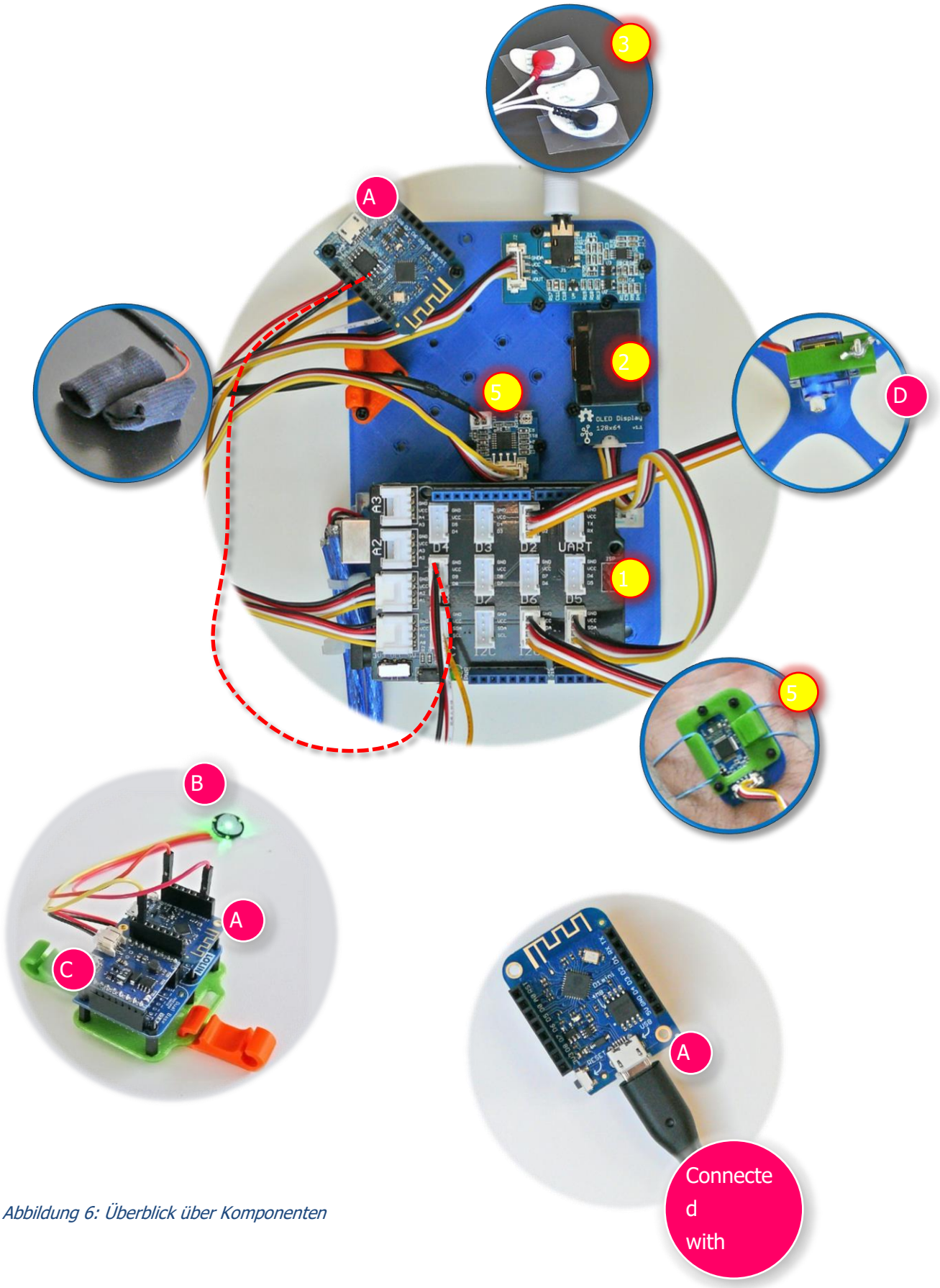


Abbildung 6: Überblick über Komponenten

Arduino-Bibliotheken für Komponenten:

Einige Komponenten wie der Wemos D1 Mini oder einige der Grove-Boards benötigen Bibliotheken, damit die Arduino IDE ordnungsgemäß funktioniert. Wie eine Bibliothek importiert wird, wird hier beschrieben:
https://www.arduino.cc/en/Guide/Libraries

OLED lib (Seeed):
https://github.com/Seeed-Studio/OLED_Display_128X64/archive/master.zip

Software I2C Master (Felias Fogg):
https://github.com/felias-fogg/SoftI2CMaster

URL for WEMOS-Boards (ESP8266):
To install the wemos, the so-called "board-definition" needs to be installed. It is described here:

http://arduino.esp8266.com/Arduino/versions/2.0.0/doc/installing.html

1. Im Fenster Arduino IDE Voreinstellungen öffnen.
2. Geben Sie die folgende URL in das Feld "Zusätzlicher Board-Manager" ein:
   http://arduino.esp8266.com/stable/package_esp8266com_index.json

3. Öffnen Sie den Board Manager aus Werkzeuge > Board Menü & suchen esp8266 Plattform.
4. Wählen Sie die aktuelle Version aus einem Dropdown-Feld aus und klicken Sie auf die Schaltfläche "Installieren".
   1. Auswählen "(LOLIN)Wemos D1 R2 und Mini" aus Tools > Board-Menü nach der Installation.

Pulsesensor:
https://pulsesensor.com/pages/installing-our-playground-for-pulsesensor-arduino

Grove GSR, Grove Finger Clip Sensor und Grove EMG Detector benötigen keine Bibliotheken, da sie mit einfachen Arduino-Befehlen gesteuert werden können.

Wemos D1 mini als drahtlose Verbindung zwischen elektronischen Komponenten

- **Wemos-Boards sollten vom Lehrer vorbereitet werden, nicht von den Schülern, bevor der Unterricht beginnt!**

- Wemos-ESP8266-Wifi-Boards sind als kostengünstigere Alternative zur zuverlässigen, aber auch teuren Xbee-Technologie gedacht.

Zwei Wemos bilden ein Paar, welchess über Wifi-Ethernet-Port 23 (das ist Telnet) verbunden ist. Der einzige Zweck besteht darin, **das serielle Kommunikationskabel** zu ersetzen. Üblicherweise wird ein experimentelles elektronisches Gerät über ein USB-Kabel mit dem PC verbunden. Um ein völlig autonomes Design zu erreichen, das nicht mit dem häuslichen Stromnetz verbunden ist, muss eine drahtlose Verbindung hergestellt werden.

Daher wird die übliche serielle Kommunikation (UART) in Wifi übersetzt und von einem Wemos, von den anderen Wemos empfangen und wieder in serielle Kommunikation übersetzt. Aus Kompatibilitätsgründen ist die Baudrate auf 9600 Baud festgelegt, da die Software-Serial-Communication durch einen Arduino Uno auf 9600 Baud beschränkt ist.

Ein Wemos D1 Mini-Paar besteht aus einem Server und einem Client. Der Server sollte mit dem PC verbunden sein. Es sollte zuerst gestartet werden und führt die folgenden Schritte aus.:

1. Scannen aller verfügbaren WLAN-Netzwerke,
2. Bestimmen, ob sich ein ungenutzter, freier Kanal oder ein schwaches Netzwerk im Hintergrund befindet,
3. Einrichtung eines Wifi Access Points über den ersten freien Kanal, auch in Kombination mit DHCP
4. Warten auf EINEN (nur einer!) Client, der eine Verbindung herstellt.
5. Wenn die Verbindung des Clients getrennt wird, wartet der Server, bis der Client erneut eine Verbindung herstellt.
6. Wenn Server zurückgesetzt wird, beginnen Sie einfach bei 1. (Scan-Netzwerke)

Der Client sollte als zweiter gestartet werden und stellt automatisch eine Verbindung und.

Konfigurieren des Wemos-Servers und -Clients, erläutert unter "Better Server sourcecode":

Hier sind die relevanten Auszüge aus Server- und Client-Sourcecode, die für die Konfiguration einzelner Wemos-Boardspaare angepasst werden müssen:

```
#include <ESP8266WiFi.h>

const char *ssid = "Erasmus";
const char *password = "12345678";
IPAddress Ip(192, 168, 3, 1);
IPAddress NMask(255, 255, 255, 0);

WiFiServer server(23);
WiFiClient serverClient;
char inChar;
```

Abbildung 7Schneiden des Server-Quellcodes

```
#include <ESP8266WiFi.h>

const char* ssid     = "Erasmus";
const char* password = "12345678";
IPAddress server(192, 168, 3, 1);

WiFiClient client;
char inChar;
```

Abbildung 8Schneiden des Client-Quellcodes

- Beide unterstrichenen Codezeilen müssen für ein Wemos-Paar genau gleich sein.
- Beide unterstrichenen Codezeilen müssen für jedes Wemos-Paar angepasst werden.

Ändern Sie die **IP-Adresse** in
192.168.1.1 OR 192.168.2.1 OR 192.168.4.1 OR 192.168.5.1 …etc.

Ändern Sie die **ssid** in
"Erasmus1" OR "Erasmus2" OR "Erasmus4" OR "Erasmus5" …etc.

… Kompilieren Sie die Skripte innerhalb der Arduino IDE und laden Sie sie auf die entsprechenden Wemos-Boards hoch.

8b Einige Theorien über Axon-Potenziale und EMG-Messung

Dieser Text ist als kurzer Überblick gedacht und kann als eine Sammlung wichtiger Schlüsselwörter betrachtet werden. Es ist nicht als Lehrbuch gedacht!

Der folgende Artikel basiert auf dem "EMG Fibel V1.1.pdf", und den Wikipedia-Einträgen zu den Schlüsselwörtern "Aktionspotenzial", "Design eines EMG-Detektors", "EMG" am "elektronischen Kompendium"”:

- http://www.velamed.com/wp-content/uploads/EMG-FIBEL-V1.1.pdf
- https://en.wikipedia.org/wiki/Action_potential
- https://iem.kug.ac.at/sid/sonic-interaction-design/forschung/hardware-software-prototyping/design-and-evaluation-of-an-emg-based-recording-and-detection-system.html
- https://www.elektronik-kompendium.de/public/schaerer/emg1.htm

Muskeln kontrahieren, weil sie elektrische Signale von Nerven erhalten: Das sogenannte "Aktionspotenzial" ist eine elektrische Potentialänderung intramuskulärer Nervenenden. Die Veränderung erfolgt zwischen -80mV ("Ruhepotential") bis +30mV ("Depolarisation"), was insgesamt eine theoretisch messbare Potentialänderung von 110mV bewirkt. Da sich Nervenenden in den Muskelfasern befinden, können wir (in der Schule) nur auf der Haut messen. Infolgedessen verschwindet ein Großteil der Aktionspotential-Signalstärke, verursacht durch den elektrischen Widerstand der Haut und des Bindegewebes. Eine typischerweise messbare Veränderung des elektrischen Potentials beträgt ca. 30mV.

Mikrocontroller, die in der Lage sind, diese Signale zu berechnen, verwenden Analog-Digital-Converter. Diese ADCs haben in der Regel einen Eingangsspannungsbereich von 0-3.3V (Wemos-Typ) oder 0-5V (Arduino Uno-Type). Die Auflösung beider ADCs beträgt 10bit, was bedeutet, dass die Mikrocontroller 3,3V vollen Messbereich in 1024 diskreten Schritten teilen können: 3,3V / 2^{10} = 3300mV / 1024 = 3,2mV. Würden wir Veränderungen des Hautpotenzials mit einem einsamen stehenden Mikrocontroller messen, könnten wir nur einen Bereich zwischen 0 und 10 von 1024 möglichen diskreten Werten erreichen. Das ist viel zu wenig. Darüber hinaus verändert sich das Hautpotenzial des menschlichen Körpers mit dem Einfluss externer elektrischer Felder, was zu potenziellen Drifts führt, die 1000-mal größer sind als die Signalstärke.

Daher benötigen wir eine vorgeschaltete elektronische Komponente, die a) unser Signal von 0,03V auf 3,3V verstärkt und b) in der Lage ist, die elektrische Felddrift zu kompensieren. Der sogenannte Instrumentationsverstärker ist eine Schaltung mit drei Eingängen: Ein Eingang für '+', einer für '-' und einer als Referenz. Während die '+' – und '-'-Elektroden' dem Zweck dienen, die 0,03V Potentialdifferenz zu messen, wird die Referenzelektrode eingesetzt, um die Drift zu kompensieren.

In diesem elektronischen Präzisionsinstrument wird das Signal verstärkt und das ist alles! Jetzt können wir das Ausgangspotenzial mit dem Mikrocontroller messen und digitalisieren.

Die Elektroden müssen sorgfältig platziert werden, da sie Nervenpotentiale messen sollten, und je kürzer der Abstand zum Nerv, desto besser das Signal: Ein Nervensignal reist mit einer Geschwindigkeit von ca. 5m/s durch den Muskel. Wir messen wir in der Mitte des Bizeps, indem beide +/- - Elektroden ca. 10cm voneinander entfernt platziert werden, während die Referenzelektrode an einem Punkt weit entfernt von dort platziert werden sollte, z.B. auf der Hand. Als erwartete Messung sollte es eine Signaltransitzeit von der Elektrode zur Elektrode von ("Depolarisationswelle") geben:

$$\frac{Abstand\ de\ Elektroden\ in\ cm}{\frac{500\ cm}{Sekunde}} = 20\ Millisekunden$$

Bei einer EMG-Hautmessung werden wir eine Überlagerung vieler Signale aus vielen verschiedenen Muskelfasern sehen und somit wird eine Wellenform von 20ms kaum zu erkennen sein. Der Wert des Aktionspotentials eines Nervs ändert sich nicht. Es gibt nur eine Änderung des Auftretens von nervösen Spannungsänderungen: Je härter ein Muskel sich zusammenziehen sollte, desto höher ist die "Feuerrate" der Aktionspotentiale der Nerven.

8c Etwas Theorie des Biofeedbacks

Dieser Text ist als kurzer Überblick gedacht und kann als eine Sammlung wichtiger Schlüsselwörter betrachtet werden. Es ist nicht als Lehrbuch gedacht!

https://en.wikipedia.org/wiki/Biofeedback

Die Reaktion eines menschlichen Körpers auf Stress oder äußere Einflüsse geschieht meist automatisch und unbewusst. Zum Beispiel, wenn ein Mensch lügt oder in Angst ist, beginnt seine Haut zu schwitzen. Dieser Schweiß kann als Änderung der elektrischen Leitfähigkeit gemessen werden, da Schweiß Elektrolyte enthält. Wenn der Messcomputer diese Veränderung visualisiert, kann der Mensch seinen emotionalen Zustand mit dem gemessenen Signal korrelieren und versuchen, seine Reaktion zu beeinflussen und zu lernen, seine Emotionen zu kontrollieren. Die vorhergehenden verborgenen Emotionen werden nun mental bewusst gemacht.

Es gibt viele Beispiele & Experimente, die Schüler selbst ausprobieren können:

- Beeinflussung der Herzfrequenz durch die Atemfrequenz, überwacht durch Pulssensoren

- Beeinflussung von Angstreaktionen durch Änderung der GSR-Aktivität durch Sensoren (Eine Angstreaktion könnte sein: ein Bild einer Spinne, ein Youtube-Video einer Achterbahn)

- Ein Polygraph (Lügendetektor) basiert unter anderem auf der Veränderung der Elektrodermalen Aktivität und kann mit GSR-Sensoren gemessen werden

- Koaktivität der Muskeln: Schreibmaschineschreiben unter Stress führt zu einer Kontraktion des Trapezmuskels beim Hals des Menschen. Dies kann mit EMG gemessen werden.

9. Unterrichtsplan: Schritt für Schritt Beschreibung der Aktivität/Der Inhalte

Lektionen 1 & 2 (90min):

Die Schüler werden am Beispiel geschult: Staubsauger-Roboter mit App-Fernbedienungen, internetbasierte Wetterstationen, intelligente Landwirtschaft und nGesundheitsanwendungen. Die Kursteilnehmer sollten untersuchen, wie diese Geräte funktionieren und welche Komponenten benötigt werden: ein Mikrocontroller-basiertes System steuert und koordiniert angeschlossene Sensoren und Aktoren. Darüber hinaus kommuniziert und koordiniert es mit anderen Systemen ähnlicher Art oft über drahtlose Kommunikationsnetze. Benötigte Teile: Sensoren, Aktoren, Kommunikationsgeräte. Möglichkeiten und Gefahren diskutieren: Wo macht IoT Sinn und wo nicht?

Lektionen 3&4 (90 min)

Biosensor-Grundlagen (Pulssensor): Den Schülern wird der erste Sensor vorgestellt, der nicht galvanisch isoliert werden muss: der Pulssensor. Es besteht aus zwei verschiedenen Teilen: einer einfachen grünen LED und einem Phototransistor plus direkt angeschlossener Verstärkerschaltung. Es muss direkt über einer Vene platziert werden, z.B. an den Fingerspitzen oder Ohrspitzen.

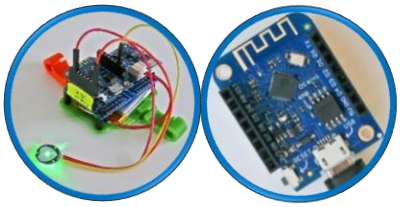


Abbildung 9: Notwendige Teile

Wenn die Vene aufgepumpt wird, da das Herz pumpt, reflektiert das Blut das grüne Licht und der Phototransistor erkennt einen hohen Wert. Wenn sich die Vene zusammenzieht, da sich das Herz zusammenzieht, führt das fehlende Blut zu einer größeren Absorption von Licht im Bindegewebe. Der Phototransistor misst einen kleineren Wert.

Die Studierenden sollten mit der analogen Programmierung von Arduino und dem analogen Wertrechner vorgestellt werden: Der Analog-Digital-Wandler (ADC) muss eingeführt werden und wie er funktioniert. Eine sehr gute Einführung finden Sie hier: https://www.generationrobots.com/media/DetecteurDePoulsAmplifie/PulseSensorAmpedGettingStartedGuide.pdf

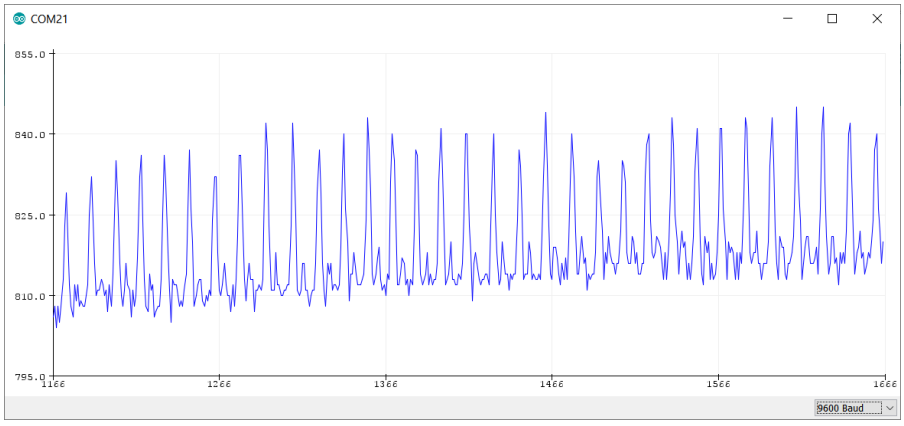
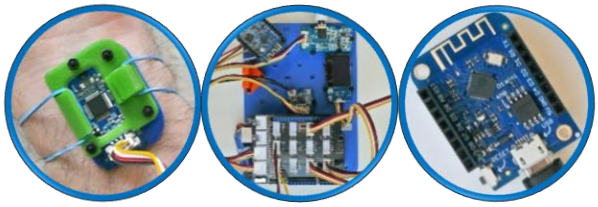


Abbildung 10: Beispieldaten des Puls-Sensors

Lektionen 5&6 (90 min)



Biofeedback der Herzfrequenz: Die Schüler lernen, dass viele Körperreaktionen emotionalen Prozessen unbewusst entsprechen. Aber wenn wir Zugang zu diesen versteckten Körperreaktionen bekommen, können wir anfangen, sie zu kontrollieren: Biofeedback ist eine Echtzeit-Überwachung der eigenen körperlichen Reaktionen mit dem Ziel, die Kontrolle über Emotionen zu erreichen.(https://www.artofmanliness.com/articles/hack-your-mind-like-a-twenty-first-century-soldier-using-biofeedback-to-become-more-resilient/).

Wenn die Pulssensor-Software richtig funktioniert, können die Schüler beginnen, Pulsänderungen zu messen und versuchen, ihre Herzfrequenz während der Atmung zu beeinflussen: Was ist der Einfluss der Atemfrequenz auf die Herzfrequenz? Was, wenn wir schneller/langsamer atmen? Was bedeutet das für Stresssituationen? Können wir unserer Herzfrequenz bewusstwerden?

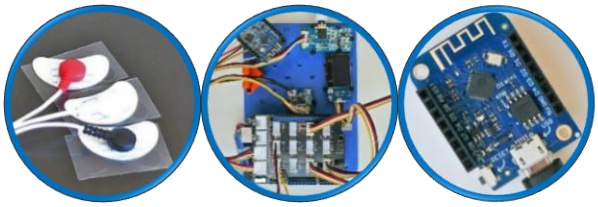
Verwenden Sie dieses Vorgehen für ein Biofeedback-Experiment:

Abbildung 9: Screenshot Processing App: Einfluss von Helligkeit / Kontrast zur Herzfrequenz

- I2C Grove Herzfrequenz Fingerclip wird an einem Finger des Schülers befestigt
- Verbinden Sie die Arduino-Station mit dem PC und starten Sie die App. Je schneller die Herzfrequenz, desto dunkler das Bild: Beeinflussen Sie Helligkeit/Kontrast mit der Kontrolle der Herzaktivität.

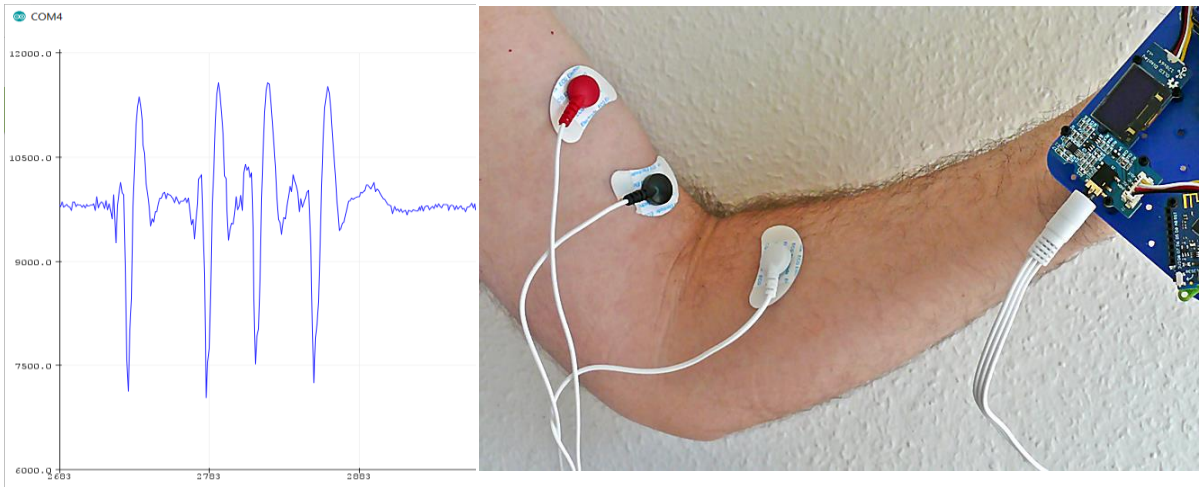
- Machen Sie nun folgendes:
    1. Atmen Sie langsam einen tiefen Atemzug für 4 Sekunden ein.
    2. Halten Sie den Atem für 4 Sekunden.
    3. Atmen Sie 4 Sekunden lang langsam aus.
    4. Halten Sie den leeren Atem für 4 Sekunden.
    5. Wiederholen Sie dies, bis Ihre Atmung unter Kontrolle ist. (https://en.wikipedia.org/wiki/Dave_Grossman_(author))

Lektionen 7&8



EMG Basics: Vermittlung der Schlüsselwörter und Grundprinzipen des aktiven Nervenpotenzials, Muskelbelastung/Entspannung und deren Messprinzipien. Die Schüler sollten verstehen, wie die Muskelsignalverstärkung funktioniert.

Wie man die drei Elektroden am Arm platziert: Zunächst muss der Testperson der Arm mit Seife und Wasser gewaschen werden und dann müssen die Stellen, an denen die Elektroden platziert werden, mit Alkohol und Watte gereinigt werden:

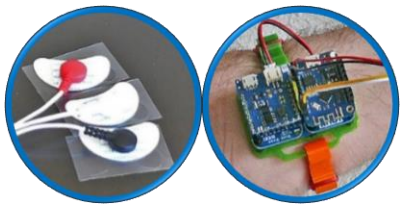


Die

Abbildung 12: Platzierung der Elektroden für Bizepsmessung

Schüler sollten mit unterschiedlichen Geschwindigkeiten der Armbewegung, unterschiedlichen Hubgewichten und der Wirkung der Entspannung experimentieren. Was passiert, wenn man mit genau dieser Elektrodenplatzierung seine Hand schließt und öffnet? Was passiert, wenn man die weiße Elektrode entfernt?

Lektionen 9&10



EMG Biofeedback Experiment:

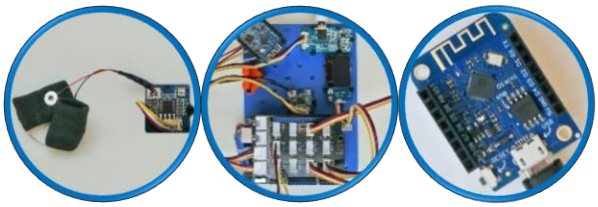
Das folgende Experiment basiert auf der Doktorarbeit von Michael Schnoz:
https://www.research-collection.ethz.ch/handle/20.500.11850/149225

Wenn eine Testperson schnell und hochkonzentriert Buchstaben auf dem Computerbildschirm eingibt, führt dies wahrscheinlich zu Schmerzen im Hals, die gemessen und anschließend von der Testperson selbst beeinflusst werden können.

Abbildung 10: Platzierung von Elektroden im Nacken

Um die Testperson unter Stressbedingungen zu stellen, könnte sie die folgende Schreib-Tutorial-Software verwenden:
https://portableapps.com/apps/education/tipp10_portable
Je schneller die Person arbeitet, desto angespannter wird die Person – und wahrscheinlich auch deren Hals. Die Testperson kann nun versuchen, diese Anspannung zu reduzieren.

Lektionen 11&12



Biofeedback GSR
https://www.youtube.com/watch?v=ZultgAFrxuc

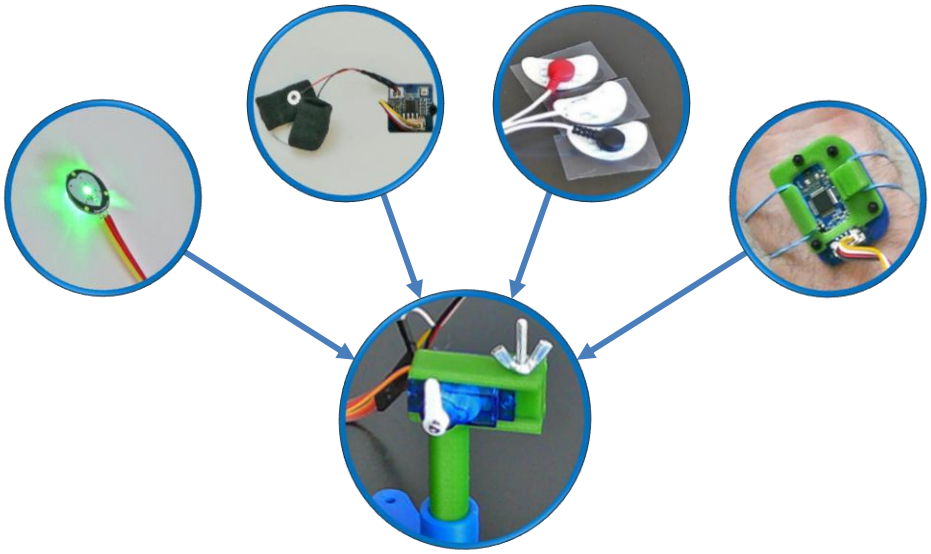
Diese Lektion basiert auf emotionalen Reaktionen beim Betrachten eines "Scary-Films": Das Auf- und Ab einer Achterbahn kann einen großen Einfluss auf die Gefühle der Probanden haben. Wie kann man es beeinflussen? Was ist mit Bildern von Spinnen oder Schlangen?

Oder von etwas Entzückendem / Angenehmem wie Musik? Wie wirkt sich Discomusik/Klassische Musik aus? Gibt es einen besonderen Effekt beim Hören Ihres Lieblingslieds?

Lektionen 13 to bis Schluss (270 min):

Freestyle Programmierung! Und: happy biofeedback! 😊

Versuchen Sie, den Ausgang der verschiedenen Sensoren mit dem Servomotor zu kombinieren. Welche Möglichkeiten haben Sie, um den Drehwinkel der Servos zu steuern??



10. Feedback	Am Ende der Lektion sollten die Schüler ein fundiertes Wissen darüber haben, wie IoT-Prinzipien in medizinischen Geräten funktionieren und wie Biofeedback helfen kann, die verborgenen Eigenschaften unseres Körpers zu verstehen. Während des Unterrichts wurden wichtige Aspekte der Elektronik, der medizinischen Informatik und der Baugrundlagen vermittelt. Darüber hinaus wurden biologische Aspekte der Muskelaktivität unterrichtet.
11. Bewertung & Evaluation	Die Schüler führen ihr Arbeitstagebuch, das vom Lehrer überprüft werden kann. Die Schüler können auch die Ergebnisse ihrer Experimente präsentieren. Darüber hinaus muss am Ende des Unterrichts ein Standard-In-Class-Test durchgeführt werden..

Wemos Client Quellcode

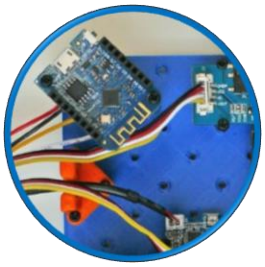
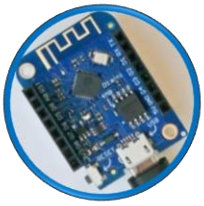


Abbildung 11Wemos-Clients an die Arduino-Station

```
#include <ESP8266WiFi.h>

const char* ssid     = "Erasmus";
const char* password = "12345678";
IPAddress server(192, 168, 3, 1);
WiFiClient client;
char inChar;

void setup() {
  Serial.begin(9600);
  WiFi.setSleepMode(WIFI_NONE_SLEEP);
  WiFi.mode(WIFI_STA);
  WiFi.setOutputPower(10);// 10: 10mW, 14: 25mW, 17: 50mW, 20: 100mW
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(5);
  }
  Serial.print("WiFi Channel: ");
  Serial.println(WiFi.channel());

  if (client.connect(server, 23))  {
    Serial.print("Local IP: ");
    Serial.println(WiFi.localIP());
    pinMode(LED_BUILTIN, OUTPUT);
    digitalWrite(LED_BUILTIN, LOW);
  }
}
void loop() {
  if (!client.connected()) {
    digitalWrite(LED_BUILTIN, HIGH);
    unsigned long startzeit = micros();
    client.connect(server, 23);
    Serial.println(micros() - startzeit);
  }else{
    digitalWrite(LED_BUILTIN, LOW);
  }

  if (client.available()) {           //wenn WIFI verfügbar ist,
    char c = client.read();           //lesen was reinkommt
    Serial.print(c);                  //und auf UART schreiben.
  }

  while (Serial.available() > 0) {    //solange auf UART Daten,...
    inChar = Serial.read();           //Daten von UART einlesen
    if (client.connected()) {         //und wenn WIFI läuft, ...
      client.write(inChar);           //auf WIFI schreiben
      delay(1);
    }
  }
}
```

Wemos Server Quellcode

Abbildung 12: Dieses Skript sollte für den mit dem PC verbundenen Wemos kompiliert werden



```
#include <ESP8266WiFi.h>

const char *ssid = "Erasmus";
const char *password = "12345678";
IPAddress Ip(192, 168, 3, 1);
IPAddress NMask(255, 255, 255, 0);

WiFiServer server(23);
WiFiClient sClient;
char inChar;

void setup() {
  Serial.begin(9600);
  unsigned int c_frei = SSID_scan();
  Serial.println("Configuring access point");
  WiFi.softAPConfig(Ip, Ip, NMask);
  WiFi.softAP(ssid, password, c_frei, false,
1);
  Serial.print("Channel: ");
  Serial.println(c_frei);

  Serial.println("Starting server");
  server.begin();
  server.setNoDelay(true);

  Serial.print("Server IP: ");
  Serial.println(WiFi.softAPIP());
  pinMode(LED_BUILTIN, OUTPUT);
  digitalWrite(LED_BUILTIN, HIGH);
}

void loop() {

  uint8_t i;
  if (server.hasClient()) {
    if (!sClient || !sClient.connected()) {
      if (sClient) sClient.stop();
      sClient = server.available();
      digitalWrite(LED_BUILTIN, LOW);
    }
  } else digitalWrite(LED_BUILTIN, HIGH);
  if (sClient.available()) {
    digitalWrite(LED_BUILTIN, LOW);
    while (sClient.available()) {
      inChar = sClient.read();
      Serial.write(inChar);
    }
  } else digitalWrite(LED_BUILTIN, HIGH);

  if (Serial.available()) {
    size_t len = Serial.available();
    uint8_t sbuf[len];
    Serial.readBytes(sbuf, len);
    if (sClient.connected()) {
      sClient.write(sbuf, len);
      Serial.write(sbuf, len);
    }
  }
}
```

```
int SSID_scan() {
  int frei = 0;
  Serial.println("scan start");
  WiFi.disconnect();
  delay(100);
  int n = WiFi.scanNetworks();
  if (n == 0) {
    Serial.println("no networks found");
    frei = 1;
  } else {
    int belegt[n];
    int staerke[n];
    Serial.print(n);
    Serial.println(" networks found.");
    for (int i = 0; i < n; ++i) {
      belegt[i] = WiFi.channel(i);
      staerke[i] = WiFi.RSSI(i);
      delay(10);
    }
    for (int i = 0; i < 12; ++i) {
      int diff = belegt[i + 1] - belegt[i];
      if (diff > 1) {
        frei = belegt[i] + 1;
        break;
      }
    }
    if (frei != 0) {
      Serial.print("done. free channel: ");
      Serial.println(frei);
      return frei;
    } else {
      int maxnummer = 0;
      int maxstaerke = staerke[maxnummer];
      for (int j = 0; j < n; j++) {
        if (maxstaerke > staerke[j]) {
          maxnummer = j;
          maxstaerke = staerke[maxnummer];
        }
      }
      frei = belegt[maxnummer];
      Serial.print("done. weakest channel: ");
      Serial.println(frei);
      return frei;
    }
  }
}
```

EMG Sourcedode Beispiel

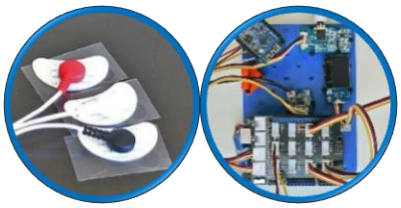


Abbildung 13: Skript für den Unterricht 7&8 an der Arduino Station

```
//Myosensor bitte an A0 anschließen
//OLED an einen der vier I2C Ports anschliessen
#include <SeeedOLED.h>
#include <Wire.h>
#include <SoftwareSerial.h>
SoftwareSerial Serial_89(8, 9);

int max_analog_dta      = 300;// max analog data
int min_analog_dta      = 100;// min analog data
int static_analog_dta   = 0;  // static analog data
int level = 5;

int getAnalog(int pin) {
  long sum = 0;
  for (int i = 0; i < 32; i++){
    sum += analogRead(pin);
  }
  Serial.println(sum);
  Serial_89.println(sum);
  int dta = sum >> 5;
  max_analog_dta = dta > max_analog_dta ? dta : max_analog_dta;
  min_analog_dta = min_analog_dta > dta ? dta : min_analog_dta;
  return sum >> 5;
}

void setup() {
  Wire.begin();
  Serial.begin(9600);
  Serial_89.begin(9600);
  SeeedOled.init();
  SeeedOled.clearDisplay();
  SeeedOled.setNormalDisplay();
  SeeedOled.setPageMode();
  SeeedOled.setTextXY(1, 0);
  SeeedOled.putString("EMG prototype");
  long sum = 0;

  for (int i = 0; i <= 10; i++) {
    for (int j = 0; j < 100; j++){
      sum += getAnalog(A0);
      delay(1);
    }
  }
  sum = sum / 1100;
  static_analog_dta = sum;
  Serial.print("static_analog_dta = ");
  Serial.println(static_analog_dta);
}

void loop() {
  int val = getAnalog(A0);
  int level2;
  if (val > static_analog_dta)  {
    level = 5 + map(val, static_analog_dta, max_analog_dta, 0, 10);
  } else {
    level = 5 - map(val, min_analog_dta, static_analog_dta, 0, 10);
  }
  for (int i = 0; i < 10; i++) {
    SeeedOled.setTextXY(1, i);
    SeeedOled.putChar(32);
  }
  for (int i = 0; i < level - 5; i++) {
    SeeedOled.setTextXY(1, i);
    SeeedOled.putChar(124);
  }
  delay(20);
}
```

Analoger Puls-Sensor am Arduino

```
//Den analogen Pulssensor bitte an A3 anschliessen:
//     _____
//    /     \
//    \  Â° /
//     | | |
//     1 2 3
// 1: Ground, 2:Plus, 3: Signal

void setup() {
  Serial.begin(9600);
  pinMode(A3, INPUT);
}

void loop() {
  unsigned int value = analogRead(A3);
  Serial.println(value);
  delay(20);
}
```

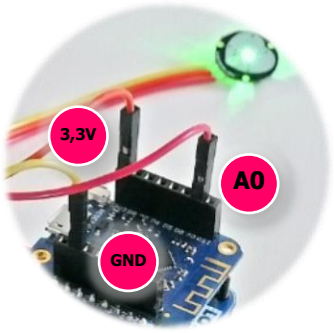


Abbildung 14: Verdrahtung Wemos und Pulssensor

Analoger Pulse-Sensor auf dem Wemos Client mit AnalogRead

```
#include <ESP8266WiFi.h>

const char* ssid     = "Erasmus";
const char* password = "12345678";
IPAddress server(192, 168, 3, 1);

WiFiClient client;
char inChar;

void setup() {
  Serial.begin(9600);
  pinMode(A0, INPUT);
  WiFi.setSleepMode(WIFI_NONE_SLEEP);
  WiFi.mode(WIFI_STA);
  WiFi.setOutputPower(10);// 10: 10mW, 14: 25mW, 17: 50mW, 20: 100mW
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(5);
  }

  Serial.print("WiFi Channel: "); Serial.println(WiFi.channel());

  if (client.connect(server, 23))  {
    Serial.print("Local IP: ");
    Serial.println(WiFi.localIP());
    pinMode(LED_BUILTIN, OUTPUT);
    digitalWrite(LED_BUILTIN, LOW);
  }
}

void loop() {
  if (!client.connected()) {
    digitalWrite(LED_BUILTIN, HIGH);
    unsigned long startzeit = micros();
    client.connect(server, 23);
    Serial.println(micros() - startzeit);
  } else {
    digitalWrite(LED_BUILTIN, LOW);
  }

  if (client.available()) {           //wenn Schnittstelle verfügbar ist,
    char c = client.read();           //lesen was reinkommt
    Serial.print(c);
  }
  delay(50);
  if (client.connected()) {         //und wenn Server-Client läuft, ...
    unsigned int value = analogRead(A0);
    Serial.println(value);
    String einsnachdemandern = String(value);
    for (int i = 0; i < einsnachdemandern.length(); i++) {
      client.write(einsnachdemandern[i]);
    }
    client.write(10);
  }
}
```

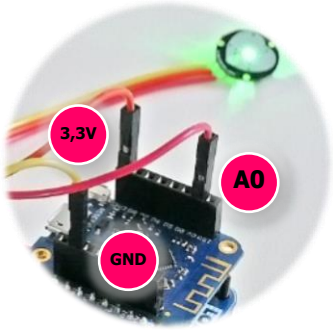


Abbildung 15: Verdrahtung Wemos - Pulssensor

GSR Quellcode an der Client-Station

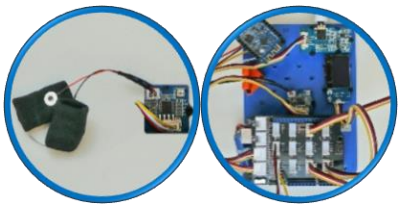
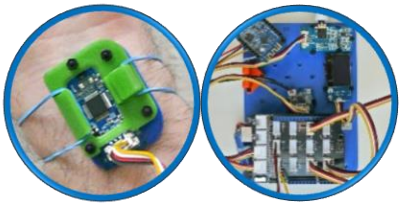


Abbildung 16 Arduino 11&12

```
//Den GSR-Sensor (fuer Elektrodermale Aktivitaet)
//bitte an A1 anschliessen
//Die OLED bitte an einen der vier I2C anschliessen.

#include <SeeedOLED.h>
#include <Wire.h>
#include <SoftwareSerial.h>
SoftwareSerial Serial_89(8, 9);

const int GSR = A1;
int sensorValue = 0;
int gsr_average = 0;

void setup() {
  Wire.begin();
  Serial.begin(9600);
  Serial_89.begin(9600);
  SeeedOled.init();
  SeeedOled.clearDisplay();
  SeeedOled.setNormalDisplay();
  SeeedOled.setPageMode();
  SeeedOled.setTextXY(1, 0);
  SeeedOled.putString("GSR prototype");
  delay(2000);
}

void loop() {
  long sum = 0;
  for (int i = 0; i < 20; i++)
  {
    sensorValue = analogRead(GSR);
    sum += sensorValue;
    delay(5);
  }
  gsr_average = sum / 10;
  Serial.println(gsr_average);
  Serial_89.println(gsr_average);
  SeeedOled.clearDisplay();
  SeeedOled.setTextXY(1, 2);
  SeeedOled.putNumber(gsr_average);
}
```

Lektion 5&6 Quellcode und Processing app

Arduino Quellcode für Grove Fingerclip Sensor:



```
//Herzraten-Sensor NICHT an I2C, sondern
//bitte an D3/D2, das ist der Port mit Aufdruck
//GND - VCC - D3 - D2
//Falls Sensor nicht anspricht, auf 3,3V stellen
#define SDA_PORT PORTD
#define SDA_PIN 3
#define SCL_PORT PORTD
#define SCL_PIN 2
#include <SoftI2CMaster.h>
#include <SoftWire.h>
#include <SeeedOLED.h>
#include <Wire.h>
#include <SoftwareSerial.h>
SoftwareSerial Serial_89(8, 9);
SoftWire SWire = SoftWire();

void setup() {
  Wire.begin();
  SWire.begin();
  Serial.begin(9600);
  Serial_89.begin(9600);
  SeeedOled.init();
  SeeedOled.clearDisplay();
  SeeedOled.setNormalDisplay();
  SeeedOled.setPageMode();
  SeeedOled.setTextXY(1, 0);
  SeeedOled.putString("heartrateprototype");
  delay(2000);
}
void loop() {
  SWire.requestFrom(0xA0 >> 1, 1);
  while (SWire.available()) {
    unsigned char c = SWire.read();
    Serial.println(c, DEC);
    Serial_89.println(c, DEC);
    SeeedOled.clearDisplay();
    SeeedOled.setTextXY(1, 2);
    SeeedOled.putNumber(c);
  }
  delay(500);
}
```

Processing Quellcode Grove Fingerclip Sensor:

```
PImage img;
int heartrate;
import processing.serial.*;
Serial ardCom;
String payload;
String[] liste;


void setup() {
  String portName = Serial.list()[0];
  ardCom = new Serial(this, portName, 9600);

  size(1000, 500);
  frameRate(20);
  img = loadImage("biofeedback.png");
  image(img, 0, 0);
  loadPixels();
}

void draw() {
  if (ardCom.available() > 0) {
    payload = ardCom.readStringUntil(,\n');
    if (payload != null) {
      heartrate = parseInt(payload.trim());
      println(heartrate);
    }
  }

  float percentage = -0.7/40*heartrate + 2.15;
  for (int x = 0; x < img.width; x++) {
    for (int y = 0; y < img.height; y++ ) {
      int loc = x + y*img.width;
      float r, g, b;
      r = red (img.pixels[loc]);
      g = green (img.pixels[loc]);
      b = blue (img.pixels[loc]);
      float bright = 64 - 64*percentage;
      float cont = percentage;
      r = r*cont - bright;
      g = g*cont - bright;
      b = b*cont - bright;
      r = constrain(r, 0, 255);
      g = constrain(g, 0, 255);
      b = constrain(b, 0, 255);
      color c = color(r, g, b);
      pixels[y*width + x] = c;
    }
  }
  updatePixels();
}
```