

Apoplexy - Ein Fitnessstracker zur Rehabilitation von Schlaganfall-Patienten?

Lukas Rost

Albert-Schweitzer-Gymnasium Erfurt

Aufbau und Schaltung des Geräts

- **Elektromyografie-Sensor:**
 - Messung der Stärke der Armmuskel-Kontraktionen
 - Messung anhand von Potentialänderungen auf der Haut
 - drei Elektroden
 - Verstärkung und Filterung
 - Ausgabe eines Analogsignals (Spannung) proportional zur Muskelaktivität
- **Mikrocontroller Atmel ATmega:**
 - Weitergabe der Sensordaten an den Bluetooth-Chip
 - 8-Bit-Mikrocontroller mit RISC-Architektur (reduzierter Befehlssatz, aber schneller)
 - Harvard-Struktur mit getrennten Speicherbereichen für Befehle und Daten
 - Schnittstellen: u.a. UART und Analog-Digital-Wandler
 - diese können über Portadressen angesprochen werden
 - gut für den Einsatzzweck geeignet (schnell, besitzt alle nötigen Schnittstellen)
- **Bluetooth-Chip HC-05:**
 - drahtlose Kommunikation mit dem Android-Mobilgerät über Funk
 - arbeitet nach dem Bluetooth-Standard
 - alle nötigen Bestandteile auf einem Chip integriert
 - Kommunikation zwischen Mikrocontroller und HC-05 per UART-Schnittstelle (digitale serielle Schnittstelle zur Datenübertragung)
- **Quarzoszillator und Kondensatoren:**
 - eingebauter Schwingquarz mit fester Frequenz
 - liefert genauen Systemtakt für störungsfreie UART-Kommunikation
 - muss über die Einstellungen (Fuses) des Mikrocontrollers angewählt werden
 - Kondensatoren für ordnungsgemäße Funktion notwendig
- **Batterie** zur Stromversorgung der Schaltung

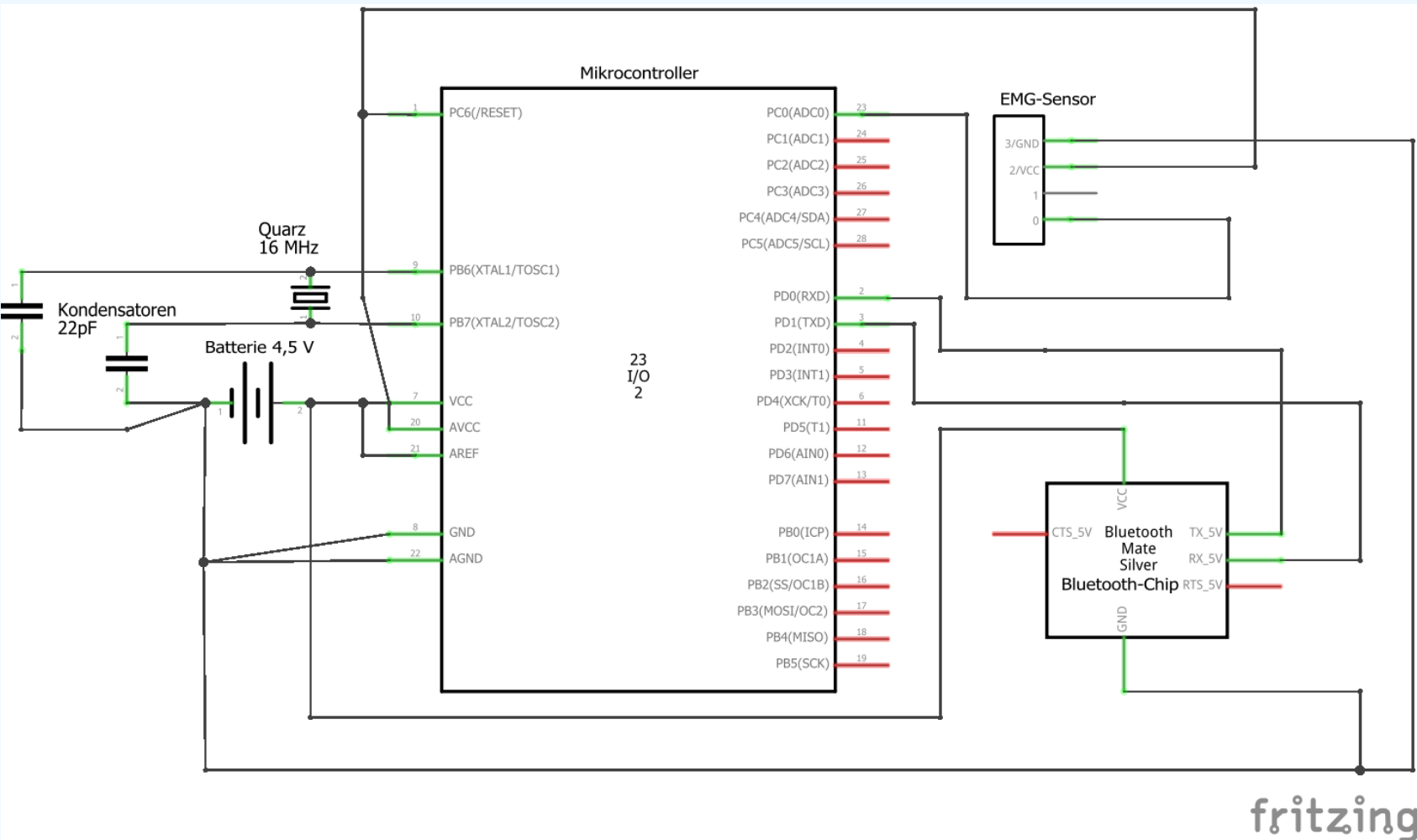


Abbildung 1. vereinfachter Schaltplan

Programmierung des Geräts

- Programm wird auf dem Mikrocontroller ausgeführt
- wurde in der Programmiersprache C programmiert
- Nutzung der AVR-Bibliotheken von Atmel
- nach dem Start: Initialisierung der Schnittstellen und Funktionen
- Senden der aktuell gemessenen Spannung über UART im Abstand einer halben Sekunde
- Messung per Analog-Digital-Wandler
- Schnittstellen werden über Register angesteuert
- Programm kann über Entwicklungsgerät in Programmspeicher (Flash) geladen werden

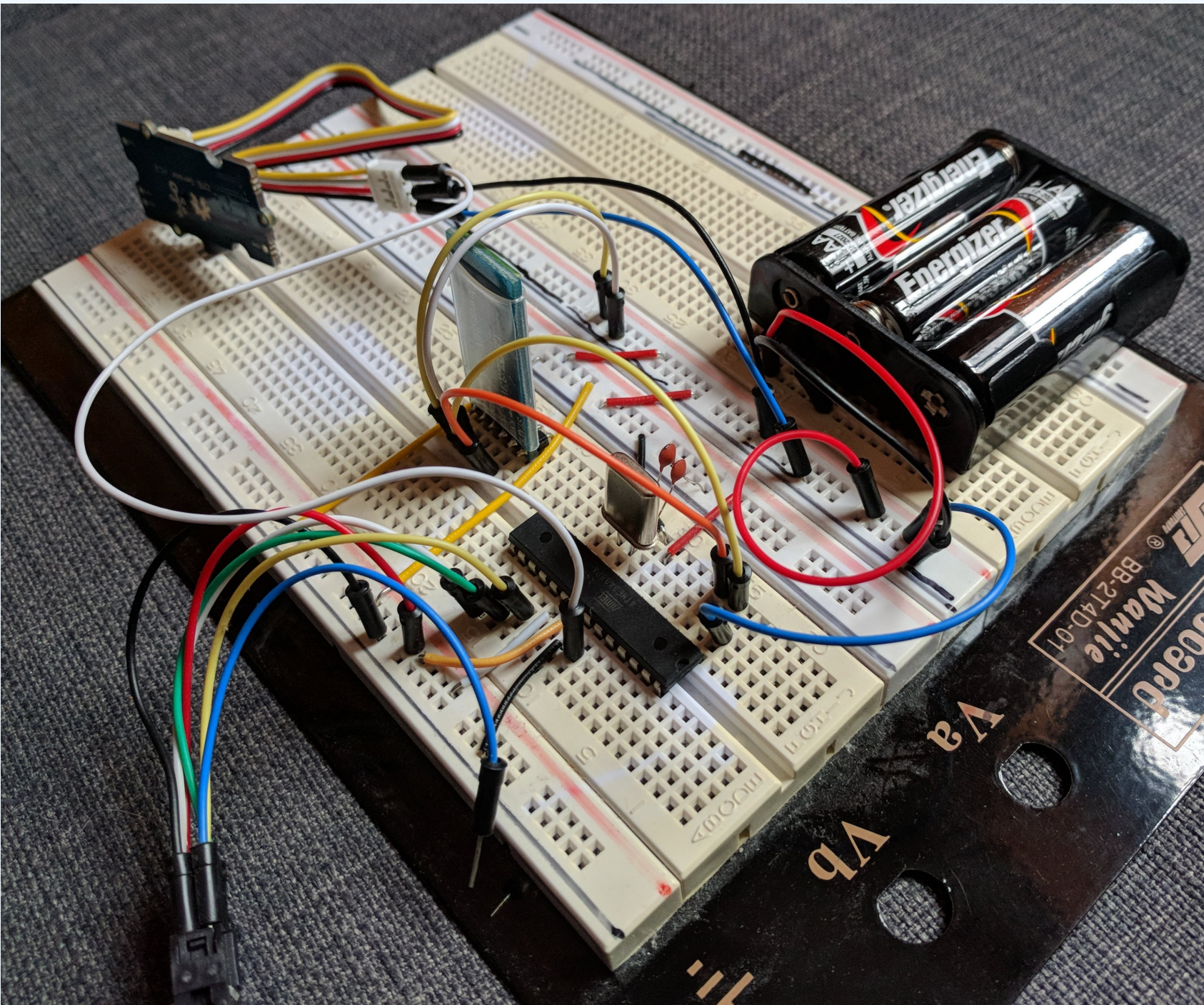


Abbildung 2. aufgebaute Schaltung auf einem Breadboard

Konzept und Aufbau der Begleitapp

- Durchführung von **Bewegungsübungen:**
 - Messwerte im zeitlichen Verlauf
 - Vergleich mit früheren Übungen
- einfaches **Minispiel** (durch Bewegungen steuerbar)
- verbindendes **Gamification-System:**
 - einfach sichtbar
 - Experience Points, Badges, Quests
- Einteilung in 4 **Bildschirmseiten** (Activities):
 - Startseite (Zusammenfassung der Erfolge, Informationstexte, Verknüpfungen)
 - Übungsseite (Darstellung der Messwerte, Ansicht der verfügbaren Quests)
 - Seite für das Minispiel
 - Einstellungsseite (u.a. für Erinnerungen)

Kommunikation mit dem Gerät

- Implementierung einer Bluetooth-Verbindung mithilfe des Android-SDK (Software Development Kit)
- Verbindung mit dem Gerät in den einzelnen Activities
- Einlesen der Bluetooth-Daten mittels einer speziellen Klasse
- Ausgleichen von Schwankungen der Messwerte durch eine Warteschlange (Queue)

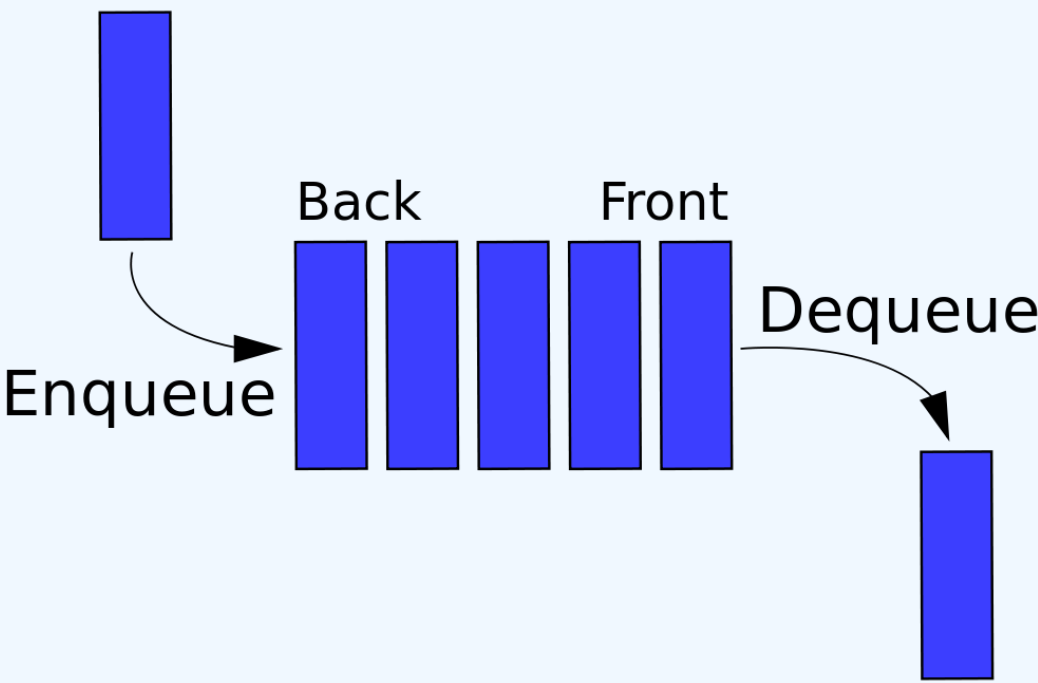


Abbildung 3. Funktionsweise einer Queue

- Berechnung eines Prozentwerts aus den Messdaten:

$$p = 100 * \frac{avg(Q) - U_{min}}{U_{max} - U_{min}}$$

Implementierung der Gamification

- **Verteilung der Erfahrungspunkte:**
 - abhängig von: Liste der prozentualen Messwerte L , Anzahl der Messwerte d
 - Punktzahl P für eine Übung:
$$P = \frac{\min(L) + avg(L) + \max(L)}{3}$$
 - Punktzahl P_S für ein Minispiel:
$$P_S = P + 2 \cdot d$$
- Quests sollen verschiedene Anforderungen zur Fertigstellung voraussetzen
- SQL-Datenbank zum Speichern dieser Anforderungen gut geeignet (SQLite)
- Aufbau der Datenbanktabelle: Titel, Beschreibung, Symbol, ...
- **mögliche Anforderungen:**
 1. bestimmte Anzahl an XP
 2. Messwert einmal während einer Übung erreichen
 3. Messwert über eine bestimmte Zeitspanne halten
- Erfolge werden für den Nutzer sichtbar gemacht (über Dialoge etc.)

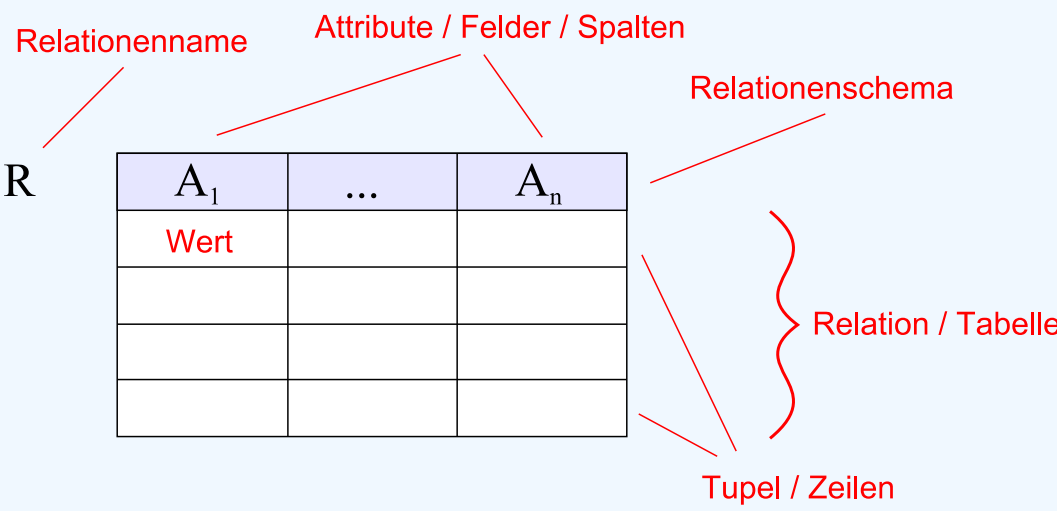


Abbildung 4. Aufbau einer relationalen Datenbank

Konzept des Minispiels

- Steuerung eines virtuellen Flugzeuges über eine Gebirgslandschaft mit Bergen unterschiedlicher Höhe
- Höhe des Flugzeugs \sim gemessene Muskelaktivität
- Ziel: Flugzeug möglichst lange fliegen lassen, ohne gegen Berg zu stoßen
- zur Implementierung genutzt: eigenes Android-Oberflächenelement (View)