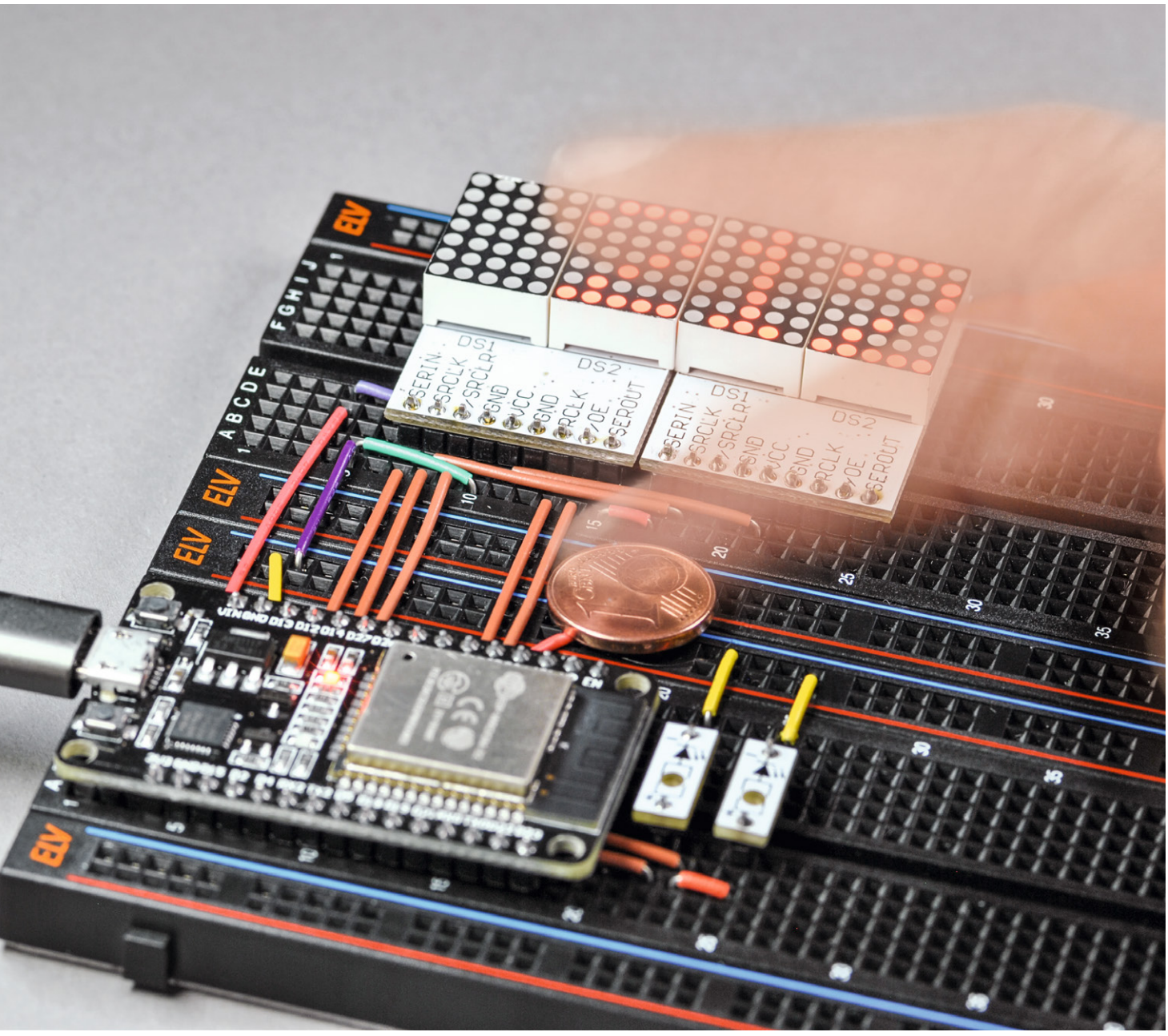
Bioelektronik IV

Reaktionsvermögen, Aufmerksamkeit und Stress

Dass das Führen von Fahrzeugen im Straßenverkehr unter Alkohol oder Drogen verboten ist, dürfte den meisten Menschen bekannt sein. Neben diesen Faktoren gibt es jedoch viele weitere Einflüsse, welche die Wahrnehmungs- und Reaktionsfähigkeit einschränken. Diese Faktoren werden häufig unterschätzt. Aber nicht nur im Verkehr, auch bei anspruchsvollen Tätigkeiten im beruflichen Umfeld ist ein gutes Reaktionsvermögen unabdingbar. In diesem Artikel zur Serie Bioelektronik sollen daher Geräte und Methoden vorgestellt werden, die es erlauben die Reaktionsgeschwindigkeit zu messen. Natürlich können die hier verwendeten Verfahren keine absolut zuverlässigen Aussagen über die Arbeitsfähigkeit oder die Verkehrstauglichkeit einer bestimmten Person zu einem gegebenen Zeitpunkt liefern. Vielmehr zeigen die Methoden einige quantitative Anhaltspunkte auf, die auf stressbedingte Erschöpfung oder allgemeine Müdigkeitserscheinungen hindeuten.



Risiken durch Übermüdung und Stress

Die Teilnahme am Straßenverkehrsgeschehen erfordert stets volle Konzentration auf die aktuelle Verkehrssituation. Auch kurze Unaufmerksamkeit kann zu folgenschweren Unfällen führen. Müdigkeit oder Alkohol-, Drogen- oder Medikamentenkonsum können das Konzentrations- und Reaktionsvermögen stark beeinträchtigen. Darüber hinaus setzen Emotionen wie Angst oder Wut die Aufmerksamkeit deutlich herab. Man geht davon aus, dass bei etwa 25 % aller schweren Unfälle Unaufmerksamkeit oder mangelndes Reaktionsvermögen im Spiel sind. Neben Alkohol gehört Müdigkeit am Steuer zu den häufigsten Unfallursachen. Insbesondere im beruflichen Fernverkehr kommt es immer wieder zu schweren Unfällen durch den berüchtigten „Sekundenschlaf". Deshalb ist es von höchster Wichtigkeit, dass Müdigkeit rechtzeitig erkannt wird und entsprechende Erholungs- oder Ruhepausen eingelegt werden.

Aber nicht nur im Straßenverkehr schränken verschiedene Substanzen die Aufmerksamkeit und Konzentration erheblich ein. Auch bei der Bedienung von Maschinen oder bei anderen komplexen Aufgaben ist die volle geistige Leistungsfähigkeit der verantwortlichen Person erforderlich. Selbst wenn es nicht zu folgenschweren Unfällen kommt, kann die Arbeit unter Alkohol-, Drogen- oder Medikamenteneinfluss zu schwerwiegenden Konsequenzen führen und die Existenzgrundlage der betreffenden Person vernichten.

Emotionen wie Angst, Ärger oder Wut bergen ebenfalls ein erhebliches Gefährdungspotenzial, vor allem wenn sie so intensiv sind, dass sie die Aufmerksamkeit einschränken oder zu unbedachten Handlungen führen.

In diesem Artikel sollen zwei Methoden vorgestellt werden, die Hinweise auf hohe Stressbelastung, reduziertes Reaktionsvermögen oder auch eine eingeschränkte Aufmerksamkeit liefern. Das erste Verfahren ist die Messung der Flimmerverschmelzfrequenz, die zweite Methode erlaubt die Bestimmung der aktuellen Reaktionszeit einer Testperson.

Flimmerverschmelzfrequenz als Ermüdungsindikator

Die Flimmerverschmelzfrequenz (FVF oder FFF für engl. flicker fusion frequency) ist diejenige Frequenz, ab der einzelne Lichtpulse nur noch als kontinuierlicher Lichtreiz wahrgenommen werden. Durch die Verschmelzung der Lichteindrücke (Flimmerfusion) wird schließlich ein einheitliches Bild ohne Helligkeitsvariation wahrgenommen.

Die FVF hängt direkt mit der Persistenz des Sehens zusammen, die für verschiedene optische Effekte oder aber auch „optische Täuschungen" verantwortlich ist. So werden Einzelbilder in rascher Abfolge als kontinuierlicher Filmablauf wahrgenommen. Bereits im 17. Jahrhundert wurde mit der sogenannten „Laterna magica" eine Reihe von Bildern, die auf eine Glasplatte gemalt waren, auf eine Wand projiziert und rasch bewegt. Dadurch konnte die Illusion „lebendiger" Bewegungen hervorgerufen werden. Erste Untersuchungen zur Flimmerfusionsfrequenz gab es

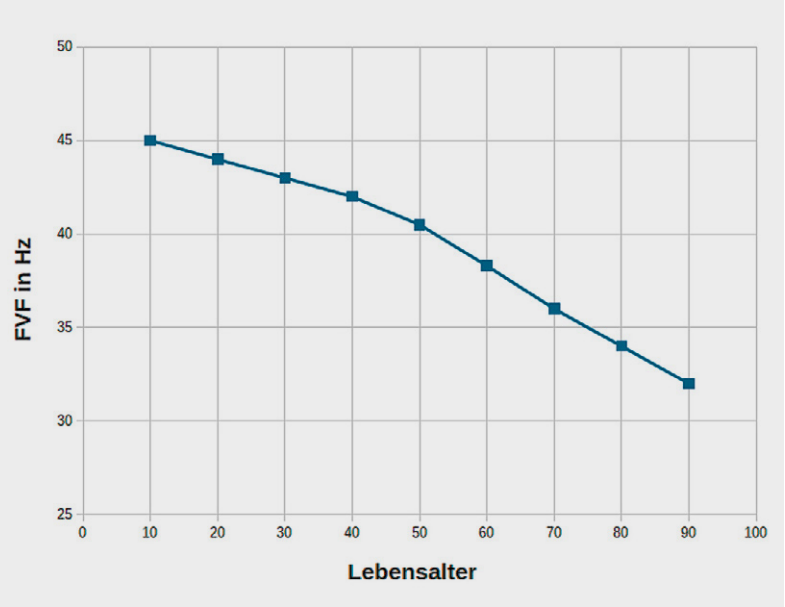


Bild 1: Abhängigkeit der FVF vom Lebensalter

dann ab dem 18. Jahrhundert. Seitdem ist bekannt, dass das Auge nicht alle Einzelheiten schneller Bewegungen erfassen kann. Die FVF hängt von mehreren Parametern ab, u. a. von:

- der Modulationstiefe des Lichtsignals
- der absoluten Beleuchtungsstärke
- der Wellenlänge des verwendeten Lichts
- der Position des Lichtreizes auf der Netzhaut
- der Dunkelanpassung des Auges
- physiologischen Faktoren wie Alter (Bild 1) und Müdigkeit

Die FVF liegt beim Menschen bei etwa 30 bis maximal 50 Einzelreizen pro Sekunde. Die Ursache für die Flimmerfusion ergibt sich daraus, dass oberhalb dieser Frequenz das Rezeptorpotenzial in den Sehzellen zwischen den Wechseln nicht mehr abklingen kann. Die einzelnen Reize verschwimmen zu einem gleichbleibenden Signalniveau, welches das Gehirn bzw. das Nervensystem als Dauerlicht interpretiert.

Da die Flimmerverschmelzfrequenz vom Ermüdungszustand abhängig ist, kann sie bis zu einem gewissen Maß als Indikator für die aktuelle Leistungsfähigkeit bzw. Erschöpfung einer Person dienen. Umfangreiche Untersuchungen zu diesem Themenkreis zeigten allerdings, dass neben dem Ermüdungszustand auch die oben genannten Einflüsse eine erhebliche Rolle spielen.

Die FVF variiert auch von Mensch zu Mensch. So soll etwa Leonardo da Vinci den maximalen Anstellwinkel von Libellenflügeln im Flug korrekt gezeichnet haben, obwohl dieser für die meisten Menschen nicht erkennbar ist. Auch verschiedene Tierarten erreichen deutlich höhere Flimmerfusionsschwellen. So wurde gezeigt, dass Tauben Werte von über 100 Hz erreichen. Auch bei Greifvögeln wurden deutlich bessere FVF-Werte gemessen als beim Menschen. Für Facettenaugen einer Fliege wurde sogar Werte von bis zu 240 Hz ermittelt.

In der Technik spielt die FVF eine entscheidende Rolle. So ist die Bildwiederholfrequenz von Fernsehgeräten oder Monitoren ein wichtiger Qualitätsfaktor. Die meisten Menschen nehmen ab einer Frequenz von ca. 75 Hz einen Bildschirm als absolut flimmerfrei war. Dieser Wert liegt also deutlich höher als die reine FVF. Dies ergibt sich aus der Tatsache, dass großflächige Lichtquellen durch die zur Erfassung notwendige Augenbewegung anders wahrgenommen werden als nahezu punktförmige Leuchtdioden. Bei der sogenannten Pulsweitenmodulation (PWM) ist die FVF ebenfalls von zentraler Bedeutung. Auch hier sollten beispielsweise die Grundfrequenzen für Sieben-Segment-Displays im Multiplexbetrieb bei mindesten 60 Hz liegen, um eine flimmerfreie Anzeige zu gewährleisten.

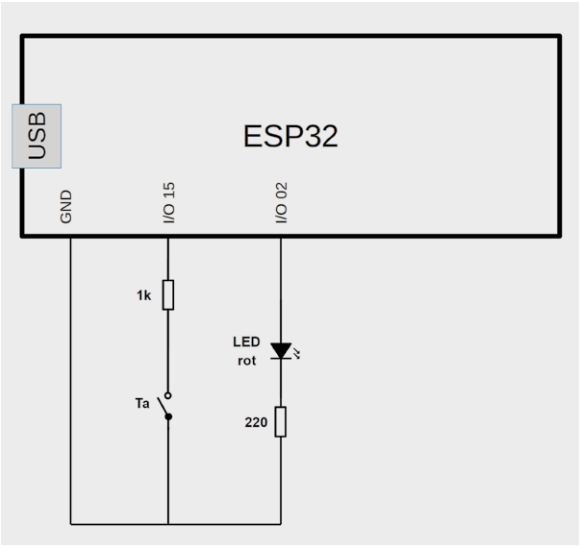


Bild 2: Schaltbild zur FVF-Messung

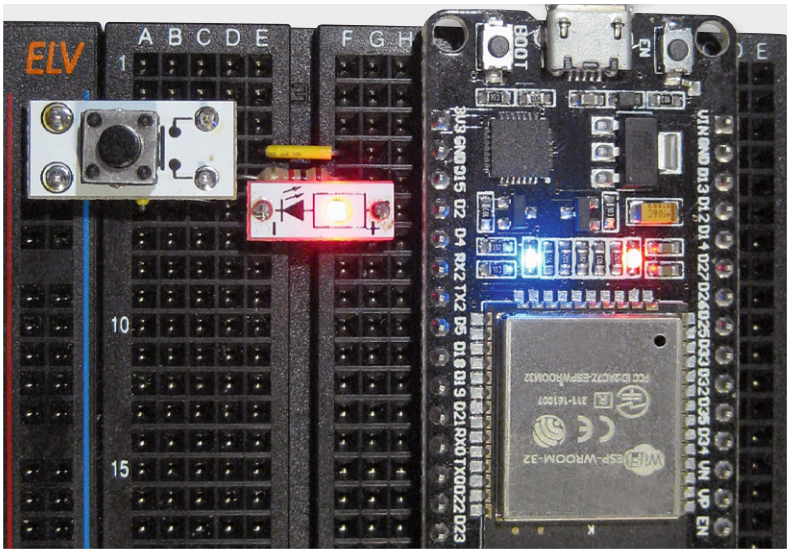


Bild 3: Aufbauvorschlag zur FVF-Bestimmung

Messung der FVF

Die individuelle FVF einer Testperson kann relativ leicht bestimmt werden. Prinzipiell sind dazu ein Rechteckgenerator und eine LED ausreichend. Lediglich die exakte Bestimmung der Blinkfrequenz ist mit einigem Aufwand verbunden. Mithilfe eines Mikrocontrollers stellt aber auch die Frequenzmessung keine unüberwindbare Hürde dar. Obwohl für diese Aufgabe auch ein wesentlich einfacherer Controller ausreichen würde, soll hierfür ein ESP32 (s. Material) zum Einsatz kommen, da dieser auch für den Reaktionstester im nächsten Kapitel verwendet werden kann. Der Hardwareaufbau ist in Bild 2 (Schaltbild) und Bild 3 (Aufbaufoto) dargestellt.

Bei einigen ESP-Boards ist der I/O-Port 2 bereits fest mit einer On-board-LED verbunden. In diesem Falle kann auf die zusätzliche Leuchtdiode verzichtet werden. Möchte man trotzdem eine externe LED verwenden, um etwa mit einer anderen Farbe zu testen, kann man auf einen anderen Port umsteigen.

Neben dem Controller selbst sind nur ein Taster und eine LED (mit Vorwiderstand) erforderlich. Beide Komponenten sind beispielsweise im PAD1- bzw. PAD2-Prototypenadapterset enthalten. Der Aufbau kann auf zwei Steckboards des Typs „ELV Breadboard schwarz" erfolgen (s. Abschnitt „Material" am Ende des Beitrags).

Der 1-kΩ-Widerstand in Serie zum Schalter dient lediglich dem Schutz des Controllers. Wird beispielsweise durch einen Softwarefehler der Port 15 als Ausgang konfiguriert und auf HIGH-Pegel geschaltet, würde ein Betätigen des Tasters ohne Serienwiderstand einen Kurzschluss verursachen. Durch den Widerstand wird der Controller auch in diesem Falle nicht gefährdet.

Softwareseitig muss ein Rechtecksignal mit einer Frequenz von 30 bis 50 Hz erzeugt werden. Dieses muss langsam von niedrigen Frequenzen zu höheren hin ansteigen oder von hohen Frequenzen beginnend abfallen. Über die Taste wird die Reaktion der Testperson abgefragt. Das folgende Programm erfüllt diese Anforderungen:

```
// FVF_test.ino
// ESP32 @ IDE 1.8.12

# define LEDpin 2
# define ButtonPin 15

int minFreq=30; // in Hz
int maxFreq=50; // in Hz
long int start;
int duration=100;  // ms per 0.1 Hertz
float peri;
bool stopIt=false;
bool InterResults=false;  // show intermediate frequencies?

void setup()
{ pinMode(LEDpin, OUTPUT);
  pinMode(ButtonPin, INPUT_PULLUP);
  Serial.begin(250000);
}

void loop()
{ // falling frequency
  for (int f=10*maxFreq; f>=10*minFreq; f--)
    { peri=5000000/f;   // halfperiode in µs
      start = millis();
```

```
      while ((millis()-start)<duration)
      { digitalWrite(LEDpin, HIGH);
        delayMicroseconds(peri);
        digitalWrite(LEDpin, LOW);
        delayMicroseconds(peri);
        if (digitalRead(ButtonPin)==LOW) stopIt=true;
        if (stopIt) break;
      }
     if (InterResults) Serial.println(f/10.0,1);
     if (stopIt)
     { Serial.print("FVF (down) = "); Serial.println(f/10.0,1);
       break;
     }
    }
    digitalWrite(LEDpin, LOW);
    delay(5000);
    stopIt=false;

  // rising frequency
  for (int f=10*minFreq; f<=10*maxFreq; f++)
    { peri=5000000/f;   // halfperiode in µs
      start = millis();
      while ((millis()-start)<duration)
      { digitalWrite(LEDpin, HIGH);
        delayMicroseconds(peri);
        digitalWrite(LEDpin, LOW);
        delayMicroseconds(peri);
        if (digitalRead(ButtonPin)==LOW) stopIt=true;
        if (stopIt) break;
      }
     if(InterResults) Serial.println(f/10.0,1);
     if (stopIt)
     { Serial.print("FVF (up  ) = "); Serial.println(f/10.0,1);
       break;
     }
    }
    digitalWrite(LEDpin, HIGH);
    delay(5000);
    stopIt=false;
}
```

Hier werden zunächst die beiden Pins für die LED-Ansteuerung (LEDpin) und die Tasterabfrage (ButtonPin) festgelegt.
Die Variablen:

```
int minFreq=30; // in Hz
int maxFreq=50; // in Hz
```

legen die Start- und die Endfrequenz fest. Die Werte können bei Bedarf angepasst werden. Wie Bild 1 zeigt, sollte ein Bereich von 30 bis 50 Hz alle Anforderungen abdecken. Ist ein schnellerer Messablauf erwünscht, kann der Frequenzbereich entsprechend eingeschränkt werden. Die Variable

```
long int start;
```

dient lediglich als Hilfswert für die Festlegung der Messzeit. Diese wird über den Wert

```
int duration=100;  // ms per 0.1 Hertz
```

festgelegt. Eine „duration" von 100 bedeutet, dass die Signalfrequenz jeweils 100 ms lang konstant bleibt und dann weiter ansteigt oder abfällt. Um eine Frequenzänderung von 1 Hz zu überstreichen, ist also eine Messzeit von einer Sekunde erforderlich. Ein Messdurchlauf (30 bis 50 Hz) erfordert damit ca. 20 s. Die halbe Periodendauer des Messsignals wird in

```
float peri;
```

gespeichert.

Als Abbruchsignal dient

```
bool stopIt=false;
```

Diese Boolesche Variable wird auf true gesetzt, sobald der Taster an Pin 15 gedrückt wird. Das Programm kann die jeweils aktuellen Frequenzwerte auf die serielle Schnittstelle ausgeben. Falls diese Zwischenwerte nicht erwünscht sind, muss

```
bool InterResults=false;  // show
intermediate frequencies?
```

auf dem Default-Wert false belassen werden. Wird die Variable auf true gesetzt, erscheinen alle Zwischenwerte im seriellen Monitor.

Im Setup werden lediglich die serielle Schnittstelle (mit 250.000 Baud) und die beiden I/O-Pins für die LED und den Taster initialisiert. Der Parameter

```
INPUT_PULLUP
```

bewirkt, dass bei dem betreffenden Port der interne Pull-up-Widerstand aktiviert wird. Damit kann auf einen entsprechenden externen Widerstand verzichtet werden.

In der Hauptschleife wird das eigentliche Messprogramm gestartet. Zunächst wird bei fallender

Frequenz gemessen. Für präzise Messungen ist eine ausreichende Frequenzauflösung erforderlich. Viele FVF-Messungen arbeiten hier lediglich mit ganzzahligen Hertz-Werten. Das hier vorgestellte Programm erlaubt dagegen eine Auflösung von 1/10 Hz. Dazu werden die Start- und Stop-Frequenzen in den For-Schleifen zunächst mit 10 multipliziert. Die Periodendauer des Signals ist der Reziprokwert der aktuellen Frequenz:

$$T = 1/f$$

Da die Variable peri die halbe Periodendauer in Mikrosekunden angibt und die Frequenzwerte verzehnfacht wurden, lautet die Umrechnung (1.000.000 µs = 1 s):

```
peri=5000000/f; // halfperiode in µs
```

Die Dauer eines Messintervalls wird über die millis()-Funktion gesteuert. Sobald die über „duration" vorgegebene Messzeit abgelaufen ist, wird der neue Frequenzwert ausgegeben. Das eigentliche Rechtecksignal wird über die klassische Port-on/off-Funktion realisiert:

```
digitalWrite(LEDpin, HIGH);
delayMicroseconds(peri);
digitalWrite(LEDpin, LOW);
delayMicroseconds(peri);
```

Die LED ist also jeweils für die Dauer des Wertes „peri" ein bzw. ausgeschaltet. Eine volle Periode entspricht daher dem Wert 2xperi (s. o.).

Die Abfragen

```
if (digitalRead(ButtonPin)==LOW)
stopIt=true;
if (stopIt) break;
```

sorgen dafür, dass der Messablauf unterbrochen wird, sobald der Taster gedrückt wurde. In diesem Fall wird die aktuelle Frequenz über

```
Serial.print("FVF (down) = "); Serial.
println(f/10.0,1);
```

auf die Konsole ausgegeben. Die zweite For-Schleife arbeitet ähnlich wie die erste, mit dem Unterschied, dass die Blinkfrequenz hier langsam ansteigt.

Nach dem Laden des Programms kann mit der Bestimmung der FVF begonnen werden. Die vollständige Messung besteht aus zwei Teilen:

1. Bestimmung der FVF von Gleitlicht zum Flimmern ...
   Die LED wird zunächst mit der Maximalfrequenz (nominal 50 Hz) moduliert. Das menschliche Auge ist nicht in der Lage, diese hohe Frequenz als Flimmern zu erkennen. Dann wird die Frequenz langsam reduziert. Sobald das erste Flimmern erkannt wird, muss der Taster gedrückt werden. Die aktuelle Frequenz wird als FVF (down) auf den seriellen Monitor ausgegeben.
2. ... und vom Flimmern zum Gleitlicht
   Nach einer kurzen Pause beginnt die LED nun mit der Minimalfrequenz von 30 Hz zu blinken. Ausgehend von diesem Wert wird die Schaltfrequenz der LED langsam gesteigert. Sobald kein Flimmern mehr erkennbar ist, wird wieder der Taster betätigt. Der aktuelle Wert erscheint nun als FVF (up) in der Konsole und die Messung startet wieder mit der abfallenden Frequenz.

Bild 4 zeigt ein entsprechendes Beispielergebnis im seriellen Monitor. Im Idealfall sollten die beiden Werte für FVF (down) und FVF (up) übereinstimmen. In der Praxis ergeben sich jedoch meist geringfügige Unterschiede. Für zuverlässige Ergebnisse sollte man die Werte mehrfach aufnehmen und mitteln. Etwa zehn Mittelungen sind im Allgemeinen ausreichend.

Nun kann man verschiedene Testpersonen überprüfen. Je nach Lebensalter oder Ermüdungszustand sollten sich unterschiedliche Resultate zeigen. Der Einfluss der Ermüdung kann durch Messungen zu verschiedenen Tageszeiten bestimmt werden. Bild 5 zeigt gemittelte Messreihen die morgens um ca. 8:00 Uhr und abends gegen 23:00 Uhr aufgenommen wurden. Jeder Messpunkt wurde über jeweils fünf Aufwärts- und fünf Abwärtswerte gemittelt.

Man erkennt, dass sich die Werte nur geringfügig unterscheiden. Für die morgendlichen Messungen ergibt sich ein Mittelwert von 42,1 Hz. Das Mittel für die abendlichen Ergebnisse liegt bei 40,8 Hz. Die Ursache für die geringe Differenz kann auch dadurch bedingt sein, dass die Tageszeit allein kein eindeutiger Indikator für den aktuellen Ermüdungszustand ist. So kann man sich durchaus am Morgen noch vergleichsweise müde, am späten Abend dagegen noch sehr fit fühlen.

Noch interessanter ist daher eventuell der Zusammenhang der Messergebnisse mit dem eigenen subjektiven Empfinden. So kann man eine Messreihe starten, wenn man sich nach einem intensiven Arbeitstag besonders abgespannt fühlt. Nach einer besonders erholsamen Nacht mit mindestens acht Stunden Schlaf sollten sich dagegen besonders gute FVF-Werte ergeben. Auch auf Partys oder Firmenfesten kann man das Gerät einsetzen. Je intensiver gefeiert wird, desto schlechter sollten die FVF-Werte werden.

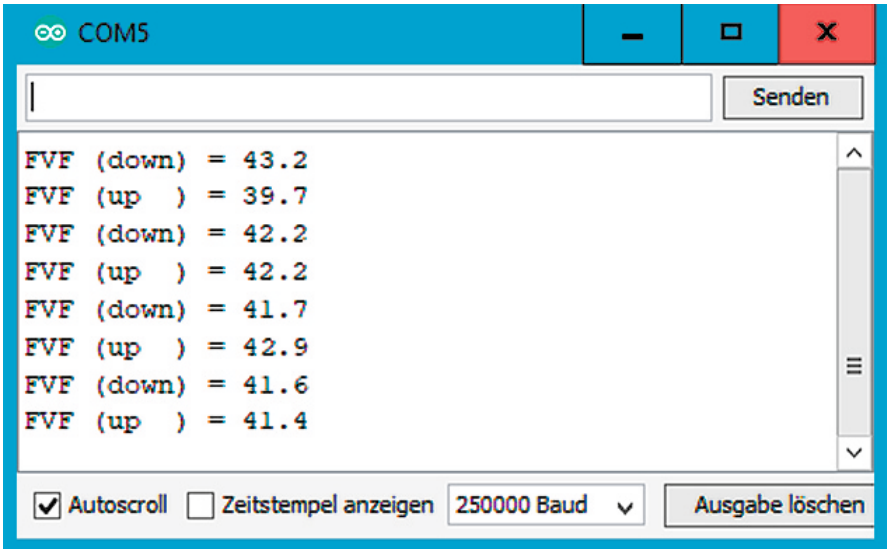


Bild 4: Messergebnisse zur FVF

Reaktionszeit, Konzentration und Aufmerksamkeit

Eine zweite Methode, um die Leistungsfähigkeit einer Person hinsichtlich Reaktionsvermögen und Aufmerksamkeit abzuschätzen, besteht in der Verwendung eines Reaktionszeitmessers. Bei der Anwendung des Gerätes geht es darum, auf eine bestimme Aktion oder Veränderung möglichst schnell zu reagieren. Plötzliche Gefahren oder Signale tauchen im Straßenverkehr immer wieder auf. Ein klassisches Beispiel ist das Aufleuchten der Bremslichter des vorausfahrenden Fahrzeugs. Hier ist eine rasche Reaktion erforderlich, wenn ein Auffahrunfall vermieden werden soll.

In einem Reaktionstester kann das Aufleuchten von Bremslichtern durch eine LED simuliert werden. Der Tester soll folgende Anforderungen erfüllen:

- plötzliche und unerwartete „Situationsänderung"
- einfache Reaktionsmöglichkeit
- Messung der Reaktionszeit zwischen Signal und Aktion der Testperson im Millisekundenbereich

Die typische Reaktionszeit eines Autofahrers auf eine unerwartete Gefahr liegt bei ca. 200 bis 300 Millisekunden (0,2 bis 0,3 s). Die Reaktionszeit ist definiert als die Zeit, die zwischen dem Auftreten eines Reizes und der Antwort auf diesen verstreicht. Dies umfasst Erkennung der Situation, ihre kognitive Verarbeitung und die korrekte Reaktion. Die Reaktionszeit hängt somit von drei verschiedenen Faktoren ab:

- Wahrnehmung: Sehen, Hören oder Fühlen eines Reizes
- Verarbeitung: Entscheidung für die angemessene Reaktion
- Antwort: Ausführung einer motorischen Aktion

Ist auch nur einer dieser Prozesse verlangsamt, wird die Reaktionszeit negativ beeinträchtigt. Da die Reaktion immer auch motorische Abläufe beinhaltet, erfordert sie ein gut ausgebildetes Reflexsystem. Von entscheidender Bedeutung ist darüber hinaus die Komplexität des Reizes. Wenn mehr Information verarbeitet werden muss, dauert der Reaktionsprozess entsprechend länger. Mehrere Faktoren beeinträchtigen die Wahrnehmung der Reaktionsreize. Unter anderem spielen

- Müdigkeit und reduzierte Aufmerksamkeit
- hohe Temperaturen
- hohes Alter
- ein zu voller Magen
- Alkohol und Drogen
- Erkrankungen und Verletzungen

eine wesentliche Rolle. All diese Faktoren haben negative Auswirkungen und können die Wahrnehmung von Reizen, ihre Verarbeitung und damit die Reaktionszeit deutlich verschlechtern.

Zu den Störungen, welche die Geschwindigkeit der Informationsverarbeitung am meisten beeinträchtigen, gehört die sogenannte diffuse axonale Verletzung. Diese tritt meist nach einer Gehirnerschütterung auf, wenn neuronale Verbindungen geschädigt werden. Ein heftiger Schlag auf den Kopf führt dazu, dass Axonen, also jener Teil der Neuronen, der die Verbindung mit anderen Gehirnzellen ermöglicht, brechen oder sogar reißen. Diese Verletzung beeinträchtigt nicht nur einen spezifischen Bereich des Gehirns, sondern alle Axonen des Gehirns. Die Folge sind diffuse Störungen in der Wahrnehmung und eine Verminderung der Verarbeitungsgeschwindigkeit. Demzufolge verlängert sich die Reaktionszeit.

Eine schnelle Reaktion ermöglicht es, verschiedene Reize und Situationen korrekt und effizient zu verarbeiten. Dies ist bei den verschiedensten Alltagssituationen wie Autofahren, in einem Gespräch, beim Sport oder bei der Bedienung von Maschinen von entscheidender Bedeutung. Durch geeignete Trainingsverfahren kann die Reaktionszeit verbessert werden. Damit können verschiedene Umstände, die zu einer geschwächten oder langsameren Reaktionszeit führen, wie Schlafmangel, negative Stimmung, Angst oder Konzentrationsmangel bis zu einem gewissen Maße ausgeglichen werden. Wird die eigene Reaktionsfähigkeit regelmäßig trainiert, werden die dadurch häufiger beanspruchten neuronalen Verbindungen stärker. Damit werden zunehmend weniger mentale Ressourcen benötigt und die Reaktionszeit verbessert sich. Dabei kann das im folgenden vorgestellte Gerät gute Dienste leisten.

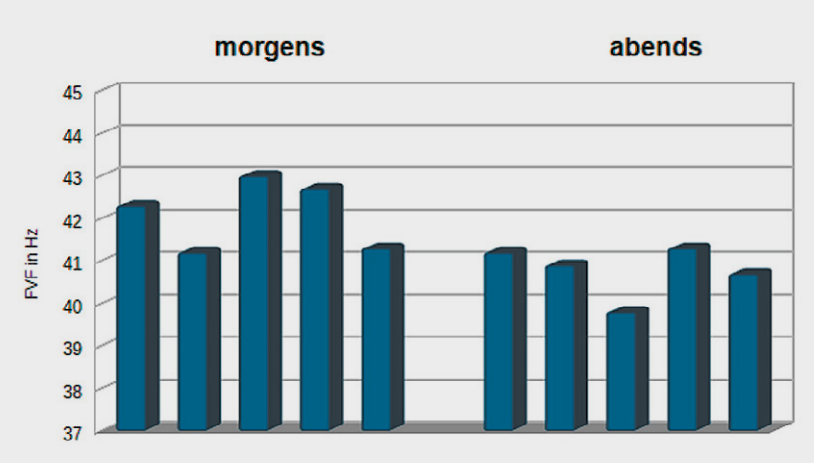


Bild 5: Abhängigkeit der FVF von der Tageszeit

Elektronischer Reaktionstester

Es existieren vielfältige Möglichkeiten für den Aufbau eines elektronischen Reaktionstesters. Die wesentlichen Komponenten sind jedoch immer eine hochauflösende Zeitmessung und die Erzeugung eines markanten Signals, auf das die Testperson in definierter Weise reagieren muss. Bei einfachen Systemen wird das Signal in regelmäßigen Zeitabständen ausgelöst. Dies verfälscht jedoch die Messung, da der Proband sich an den fest vorgegebenen Zeitraum gewöhnt. Dieser Einfluss kann eliminiert werden, wenn das Signal in zufälligen Zeitabständen erscheint. Mithilfe eines Mikrocontrollers ist die Erzeugung von zufälligen Zeitintervallen problemlos möglich. Dass es sich dabei lediglich um Pseudo-Zufallsabstände handelt, spielt hier keine wesentliche Rolle. Die Varianz der Pseudo-Zufallszahlen kann problemlos so gewählt werden, dass ein Gewöhnungseffekt vollständig unterbunden wird.

Die Basis bildet ein ESP32-Modul. Für die Anzeige der Reaktionszeit werden zwei LED-Matrixmodule verwendet. Diese sind im Prototypenadaptersatz PAD4 (s. Materialliste) enthalten. Die Ansteuerung der Module erfolgt über Schieberegister. Dazu werden fünf Signalleitungen benötigt. Diese Verbindungen sind in Tabelle 1 zusammengefasst.

Da zwei Module eingesetzt werden, sind diese über die Datenleitungen zu verbinden, d. h. SEROUT des ersten Moduls muss mit SERIN der zweiten Einheit verdrahtet werden. Nach dem Anschluss der beiden LED-Punktmatrizen kann der Aufbau mit dem Sketch „DotMatrixCounter.ino" (s. Downloadpaket [1]) getestet werden. Nach dem Laden des Sketches sollte die Anzeige etwa im Sekundentakt von 0 auf 9999 zählen.

Zur Signalisierung werden zwei LEDs in den Farben rot und grün verwendet. Die rote LED leuchtet im Ruhezustand. Sobald diese erlischt und die grüne

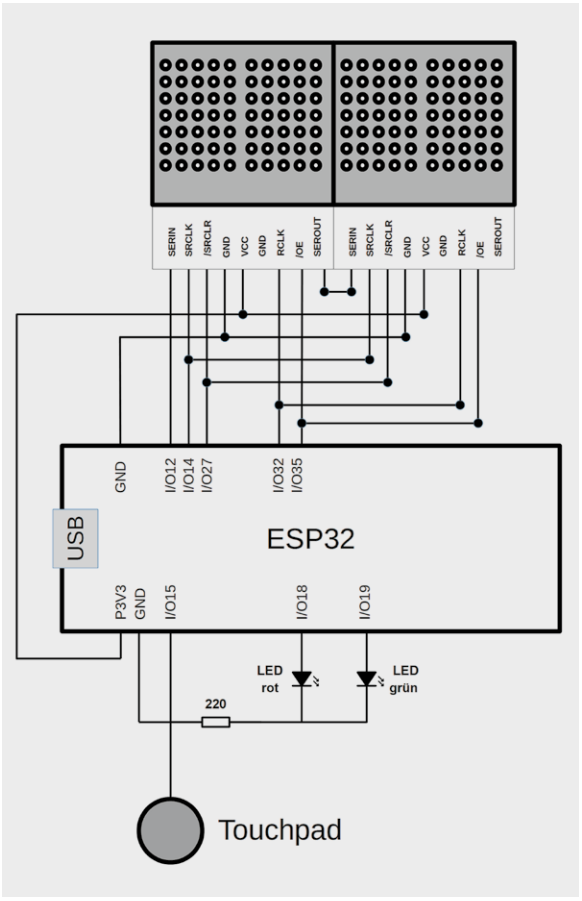


Bild 6 : Schaltung zum Reaktionsmessgerät

Leuchtdiode aktiv wird, ist eine schnelle Reaktion gefragt. Dies entspricht beispielsweise einem möglichst schnellen Start an einer Formel-1-Ampel. Möchte man eher die Reaktion auf das Aufleuchten eines roten Bremslichtes simulieren, kann man die grüne LED gegen eine rote tauschen. Die ursprüngliche rote LED entfällt in diesem Falle. Das „Startsignal" ist dann das Aufleuchten der roten LED.

Die geforderte Reaktion besteht aus der Berührung eines Touch-Sensors. Die Zeit zwischen dem Umschalten der LEDs und der Sensorberührung wird gemessen und in Millisekunden auf den beiden Punktmatrixanzeigen dargestellt. Die Auflösung von einzelnen Millisekunden stellt für den ESP-Controller kein Problem dar. Zudem ist sie bei einer zu erwartenden Reaktionszeit von 200 bis 300 ms vollkommen ausreichend. Der Touch- bzw. Berührungssensor bietet sich an, da der ESP32 über zehn integrierte Touch-Sensoreingänge verfügt. Damit kann auf die ansonsten üblichen Taster verzichtet werden.

Steuerung über Berührungssensoren

Die zehn Touch-Sensor-Eingänge des ESP32 liegen an den folgenden I/O-Ports:

Touch0 - GPIO4	Touch1 - GPIO30 (n. v.)
Touch2 - GPIO2	Touch3 - GPIO15
Touch4 - GPIO13	Touch5 - GPIO12
Touch6 - GPIO14	Touch7 - GPIO27
Touch8 - GPIO33	Touch9 - GPIO32

Touch1 ist auf dem Node-MCU-Board nicht verfügbar (n. v.). Alle anderen Sensoreingänge sind frei verwendbar. Für den Reaktionstimer wurde GPIO15 (Touch3) gewählt. Die Ansteuerung der Touch-Sensoren erfordert lediglich ein geeignetes „Pad". Dieses kann beispielsweise aus einem Messingreißnagel oder aus einem geeigneten Stück Kupferblech mit Drahtanschluss o. Ä. bestehen. Den zugehörigen Aufbau zeigt Bild 7.

Die Funktion der Touch-Sensoren kann mit dem Sketch „TouchTest.ino" überprüft werden (s. Downloadpaket). Nach dem Laden des Sketches wird der serielle Monitor der Arduino IDE gestartet. Wenn nun das TouchPad berührt wird, fällt der ausgegebene Messwert deutlich ab. Das Ergebnis im seriellen Plotter sollte ähnlich aussehen wie in Bild 8.

Zudem sollte beim Berühren des Pads die an Port 18 angeschlossene LED aufleuchten. Wie Bild 8 zeigt, liefert der Touch-Sensor im Ruhezustand Werte um 65. Sobald der Sensor berührt wird, fallen diese auf ein Niveau zwischen 8 bis 9 ab. Mit einem Schwellwert von

```
threshold=15;
```

ergibt sich also eine sichere Erkennung des Sensorsignals. Falls sich bei dieser Messung andere Werte ergeben, kann die Schwelle im Sketch (threshold) entsprechend angepasst werden. Es empfiehlt sich, die Schwelle eher niedrig anzusetzen, da sich eventuelle Störeinstreuungen vor allem auf den Ruhezustand des Sensors auswirken. Da sich insbesondere in elektromagnetisch stark belasteten Umgebungen (WLAN, Smartphones, Elektromotoren etc.) stärker schwankende Werte ergeben können, wurde der Sensorwert im Sketch 10-fach gemittelt. Im Bedarfsfall kann auch die Anzahl der Mittelungen variiert werden.

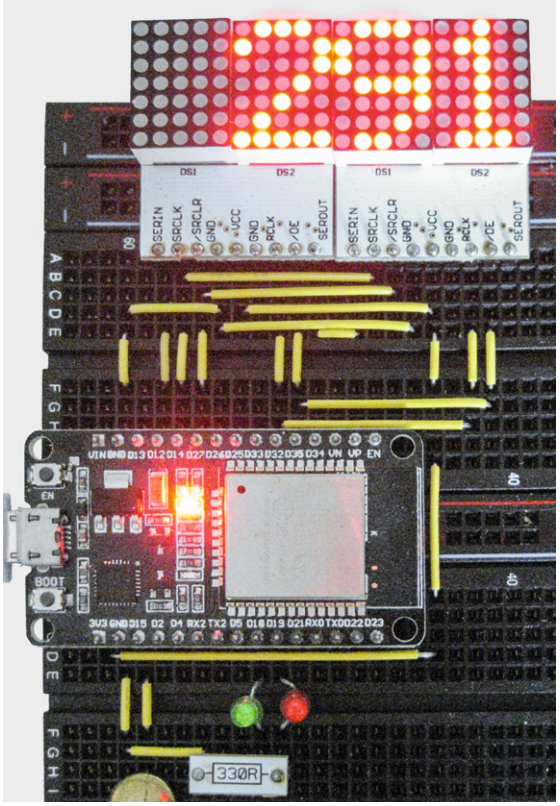


Bild 7: Der Reaktionstester in Aktion: Reaktionszeit: 291 ms = 0,291 s

Tabelle 1

Verbindungen der Signalleitungen der LED-Matrixmodule zum ESP32

Funktion	Bezeichnung am Modul	Pin am ESP32
SERIAL DATA IN	SERIN	12
SERIAL_CLOCK	SRCLK	14
SERIAL CLEAR	/SRCLR	27
REGISTER_CLOCK	RCLK	32
OUTPUT ENABLE	/OE	35

Wenn die Funktion des Touch-Sensors erfolgreich getestet wurde, kann der Sketch des Reaktionszeitmessers geladen werden. Das vollständige Listing dazu sieht so aus:

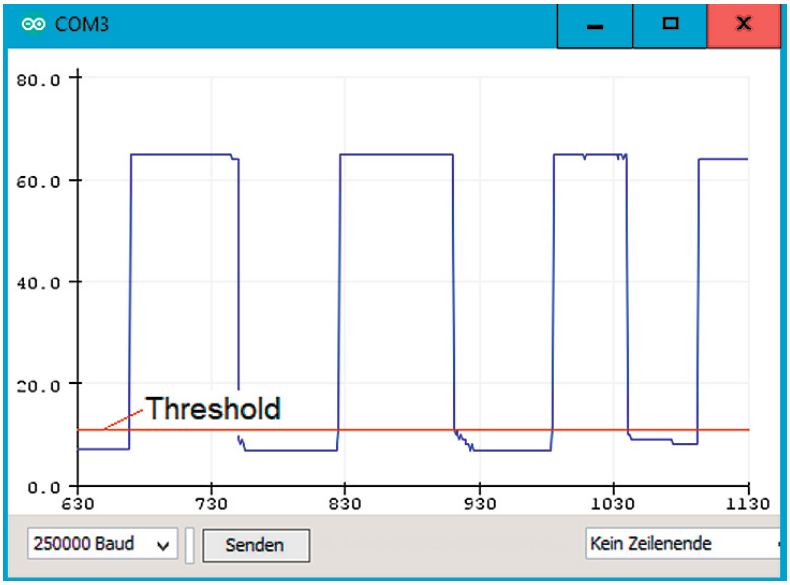


Bild 8: Signalverlauf beim Berühren des Touch-Sensors

```
// Reaction_TouchTimer.ino
// ESP32 @ IDE 1.8.12

#include "PAD4_DM.h"
#define NUMBER_OF_MODULES   2
#define SERIAL_DATA_PIN     12
#define SERIAL_CLOCK_PIN    14
#define SERIAL_CLEAR_PIN    27
#define REGISTER_CLOCK_PIN  32
#define OUTPUT_ENABLE_PIN   35

#define TOUCH_PIN           15
#define LED_RED             18
#define LED_GREEN           19

PAD4_DM Display_Module;
const byte threshold=15;       // adapt to set-up
char strng [5];    // 4 digits + "/0"

byte minTime=10, maxTime=50; // time delay boundaries in 1/10 secs
int randomDelay;
int touchAverages=10;
int touchReadVal;
int start, reactionTime;

void setup()
{ Display_Module.begin(NUMBER_OF_MODULES, SERIAL_DATA_PIN, SERIAL_CLOCK_PIN, SERIAL_CLEAR_PIN,
REGISTER_CLOCK_PIN, OUTPUT_ENABLE_PIN );
    pinMode(LED_RED, OUTPUT); digitalWrite(LED_RED, LOW);
    pinMode(LED_GREEN, OUTPUT); digitalWrite(LED_GREEN, LOW);
    randomSeed(100);
    Serial.begin(250000);
}

void loop()
{  digitalWrite(LED_RED, HIGH);
   randomDelay=random(minTime, maxTime);
   for (int i=0; i<randomDelay; i++)
   { touchReadVal=getTouchValue(TOUCH_PIN, touchAverages);
     Serial.println(touchReadVal);
     if(touchReadVal<threshold) disqualified(1000);
     delay(100);
   }
   digitalWrite(LED_RED, LOW); delay(10);
   digitalWrite(LED_GREEN, HIGH); delay(10);
   start=millis();
   while (getTouchValue(TOUCH_PIN, touchAverages)>=threshold);
   reactionTime=millis()-start;
   digitalWrite(LED_GREEN, LOW);
   Serial.println(reactionTime);
   if (reactionTime>9999) overflow(3000);
   else showNumber (reactionTime,3000);
}

void showNumber(int N, int duration)
{ int start=millis();
  while((millis()-start)<duration)
  { sprintf(strng,"%4u",N);
    Display_Module.set_display_buffer(strng);
    Display_Module.update_matrix();
  }
  Display_Module.set_display_buffer("    ");
  Display_Module.update_matrix();
}
```

```
void disqualified(int duration)
{ int start=millis();
  while((millis()-start)<duration)
  { Display_Module.set_display_buffer("DISQ");
    Display_Module.update_matrix();
  }
  Display_Module.set_display_buffer("    ");
  Display_Module.update_matrix();
}

void overflow(int duration)
{ int start=millis();
  while((millis()-start)<duration)
  { Display_Module.set_display_buffer("OVFL");
    Display_Module.update_matrix();
  }
  Display_Module.set_display_buffer("    ");
  Display_Module.update_matrix();
}

int getTouchValue(int PinNumber, int averages)
{ int touchVal=0;
  for (int i=0; i<averages; i++) touchVal+=touchRead(PinNumber);
  touchVal/=averages;
  return touchVal;
}
```

Zunächst wird die Bibliothek für die Ansteuerung der Dot-Matrix-Module eingebunden. Die zugehörige Library kann unter [2] von der ELV-Internetseite heruntergeladen werden. Dann werden die Pin-Nummer für die Dot-Matrix-Displays, den Touch-Sensor und die zwei LEDs (rot und grün) festgelegt. Anschließend wird das Display initialisiert, der Threshold für den Touch-Sensor festgelegt und der Hilfsstring für die LED-Matrix definiert.

Es folgt die Festlegung der Steuerparameter. Über

```
byte minTime=10, maxTime=50; // time delay
boundaries in 1/10 secs
```

können die Grenzen der Wartezeit zwischen den einzelnen Ereignissen in 1/10 s eingestellt werden. Mit der Voreinstellung beträgt die minimale Zeit zwischen zwei Umschaltvorgängen 1 s, die maximale Wartezeit liegt bei 5 s.

Im Setup werden die Dot-Matrix, die LED-Ports, der Touch-Sensor-Eingang und die serielle Schnittstelle initialisiert. Mit randomSeed() wird der Pseudozufallszahlengenerator gestartet. Diese Zufallssequenz ist zwar sehr lang, jedoch stets dieselbe. In dieser Anwendung spielt das keine Rolle, da die Ereignisse mit ausreichender Varianz erzeugt werden, um Gewöhnungseffekte zu vermeiden.

In der Hauptschleife wird zunächst die rote LED eingeschaltet. Damit ist der Reaktionszeitmesser in Bereitschaft. Dann wird über

```
randomDelay=random(minTime, maxTime);
```

eine zufällige Verzögerungszeit erzeugt. Die folgende For-Schleife sorgt dafür, dass die Schaltung über die Wartezeit hinweg in Bereitschaft bleibt. Wird während dieser Zeit der Touch-Sensor aktiviert:

```
touchReadVal=getTouchValue(TOUCH_PIN,
touchAverages);
Serial.println(touchReadVal);
if(touchReadVal<threshold) disqualified(1000);
```

wird auf der Matrixanzeige „DISQ“ für Disqualifikation angezeigt. Damit wird verhindert, dass durch permanentes Berühren des Sensors falsche Werte erzeugt werden.

Ist die zufällige Wartezeit abgelaufen, wird die rote LED ausgeschaltet und die grüne aktiviert:

```
digitalWrite(LED_RED, LOW); delay(10);
digitalWrite(LED_GREEN, HIGH);
delay(10);
```

Dies ist das Startsignal, auf das nun möglichst schnell reagiert werden muss. Gleichzeitig wird die Startzeit via millis()-Funktion erfasst. Die Zeit läuft so lange, bis der Touch-Sensor berührt wird:

```
while (getTouchValue(TOUCH_PIN,
touchAverages)>=threshold);
```

Anschließend wird die Reaktionszeit als Differenz zwischen Start- und Stoppzeit berechnet und auf das Display und die serielle Schnittstelle ausgegeben. Die Zeile

```
if (reactionTime>9999)
overflow(3000);
```

sorgt dafür, dass bei Reaktionszeiten von mehr als 9999 ms die Anzeige „OVFL“ für overflow, also Überlauf ausgegeben wird. Nachdem die grüne LED wieder deaktiviert wurde, steht der Reaktionstester für eine neue Messung zur Verfügung. Die drei Unterroutinen

```
void showNumber(int N, int duration)
void disqualified(int duration)
void overflow(int duration)
```

sorgen dafür, dass die gemessene Zeit bzw. die jeweils korrekten Informationen an das Display ausgegeben werden. Die Routine

```
int getTouchValue(int PinNumber,
int averages)
```

liefert den aktuellen Messwert des Touch-Sensors am Pin „PinNumber“ zurück. Der Parameter „averages“

legt die dabei verwendete Anzahl der Mittlungen fest. Der Parameter „duration" gibt an, wie lange das Display jeweils aktiv bleibt.

Reaktionstraining

Wenn sowohl das Dot-Matrix-Display als auch der Touch-Sensor erfolgreich getestet wurden, kann der Sketch „Reaction_TouchTimer.ino" auf den ESP32 geladen werden. Damit ist der Reaktionstester einsatzbereit. Nun kann die Reaktionszeit von Freunden, Bekannten oder Familienmitgliedern gemessen werden. Hier bestätigt sich meist der Trend, dass ältere Personen längere Reaktionszeiten benötigen. Bild 9 zeigt eine typische Verteilung für verschiedene Testpersonen:

- Testperson A: männlich 52 Jahre
- Testperson B: weiblich 48 Jahre
- Testperson C: weiblich 16 Jahre
- Testperson D: weiblich 22 Jahre
- Testperson E: männlich 78 Jahre

Zudem kann man die Reaktionsfähigkeit bei verschiedenen Ermüdungszuständen bestimmen. Auch hier bestätigt sich häufig, dass die Zeiten mit zunehmender Ermüdung länger werden (Bild 10).

Im Gegensatz zur FVF lässt sich die Reaktionszeit bis zu einem gewissen Maße durch spezielle Übungen verbessern. Durch intensives Training können Reaktionszeiten um bis zu 30 % reduziert werden. Dies ist ein Hinweis darauf, dass die neuronale Verflechtung im Gehirn für diese spezielle Aufgabe verbessert wird. Inwieweit sich auf diese Weise die allgemeine Reaktionszeit einer bestimmten Person beeinflussen lässt, ist umstritten. Bild 11 zeigt jedoch einen klaren „Lernerfolg" bei Anwendung des Reaktionszeitmessers.

Auch bei längerfristigem Training über einen Monat hinweg zeigen sich gewisse Erfolge (Bild 12). Diese sind jedoch aufgrund der relativ geringen Anzahl von Messpunkten nicht zweifelsfrei statistisch signifikant.

Fazit und Ausblick

Subjektive Phänomene wie Müdigkeit oder Aufmerksamkeit können mit elektronischen Mitteln bis zu einem gewissen Maße quantifiziert werden. So zeigen sich sowohl bei der Reaktionszeit als auch bei der FVF mehr oder weniger deutliche Korrelationen zum subjektiven Empfinden oder zur Tageszeit.

Die hier vorgestellten Schaltungen erlauben die Bestimmung von Parametern, die für gewisse Tätigkeiten wie etwa Autofahren oder Maschinenbedie-

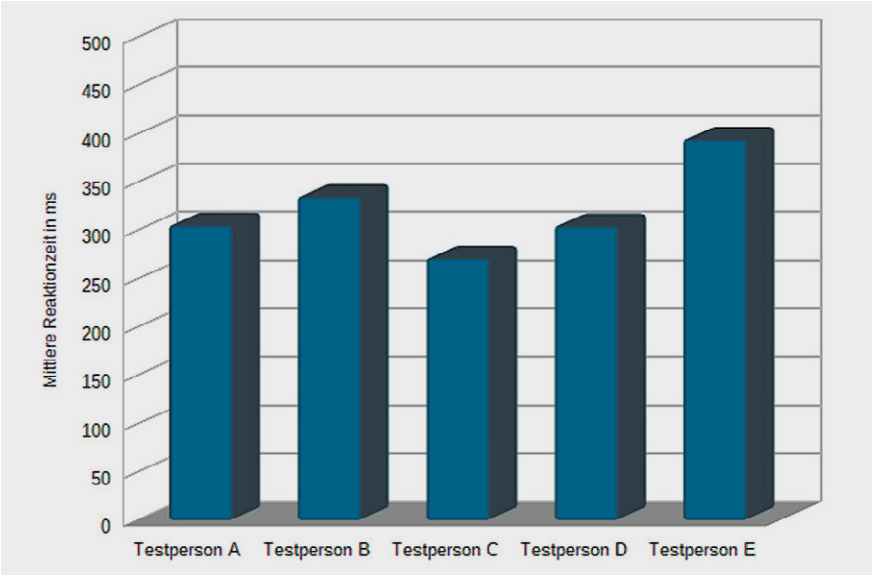


Bild 9: Typische Reaktionszeiten verschiedener Testpersonen

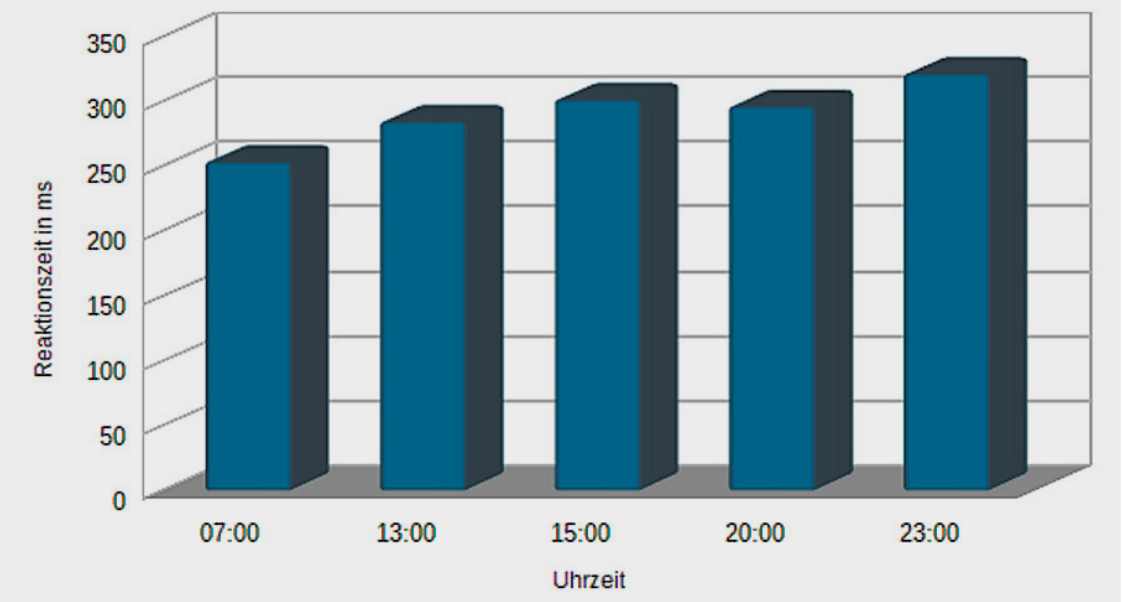


Bild 10: Zunahme der Reaktionszeit mit der Ermüdung bzw. Tageszeit

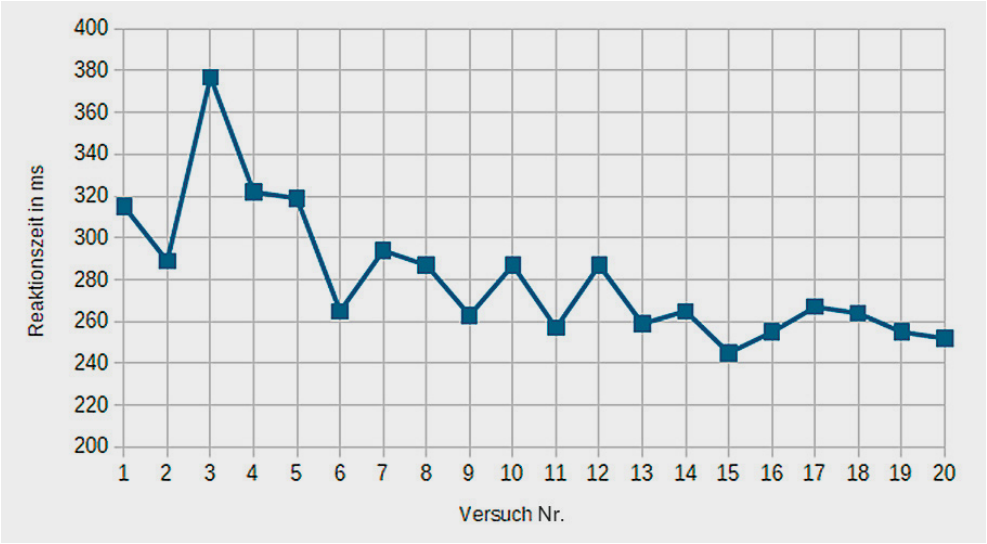


Bild 11: Kurzfristiger Lernerfolg

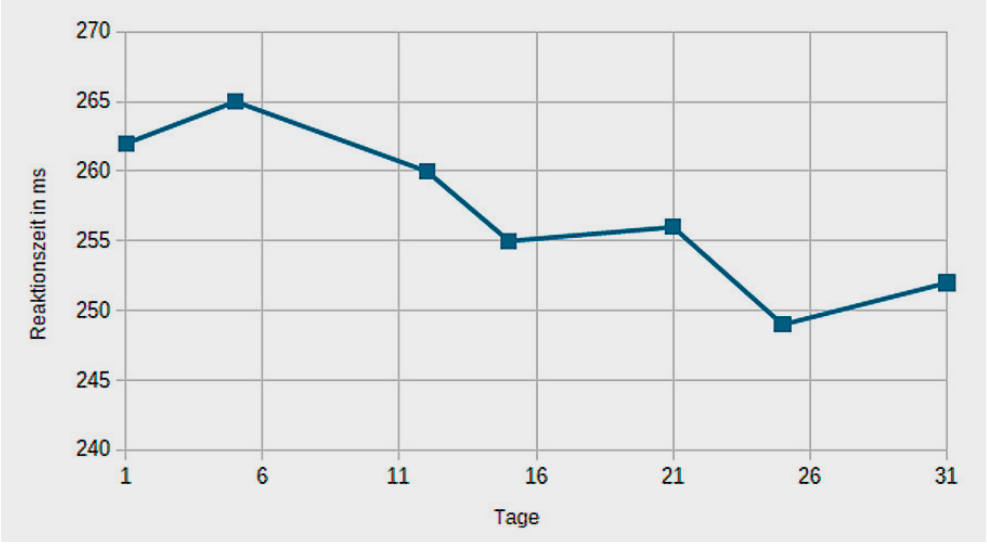


Bild 12: Verbesserung der Reaktionszeit durch Langzeittraining

nung von entscheidender Bedeutung sind. Natürlich können die Methoden nicht für die zuverlässige Bestimmung der Fahrtüchtigkeit herangezogen werden. Für eine gewisse Einschätzung im Rahmen des Familien- oder Freundeskreises sind die Geräte aber dennoch aufschlussreich. Bei bestimmten Gelegenheiten wie Partys oder Festen haben sie zudem einen nicht unerheblichen Unterhaltungswert. Auch für Trainingszwecke und zur Selbsteinschätzung sind sie durchaus sinnvoll einsetzbar. Insbesondere umfangreichere Datensätze können sogar Aufschlüsse über das persönliche Lernverhalten bezüglich eines eigenen Reaktionstrainings liefern.

Auch im nächsten Artikel zu dieser Reihe wird es um bioelektronische Anwendungen gehen, die für Trainingszwecke eingesetzt werden können. Im Vordergrund steht dabei die Erfassung der elektrischen Herzsignale. Ein EKG-Verstärker wird dabei die vollständige und detaillierte Aufzeichnung eines Elektrokardiogramms ermöglichen. Die Erfassung der Herzaktivitäten über einen längeren Zeitraum hinweg ermöglicht es unter anderem, den eigenen Fitness- und Trainingszustand einzuschätzen. ELV

Weitere Infos:

[1] Downloadpaket zum Beitrag: Artikel-Nr.: 251599

[2] Bibliothek für die Ansteuerung der Dot-Matrix-Module finden Sie im ELVshop bei der Artikelbeschreibung im Download-Bereich unter der Artikel-Nr. 155107

Ein Video, das den Reaktionszeitmesser in Aktion zeigt, kann unter https://www.youtube.com/watch?v=wGsE18P6NMo abgerufen werden.

Alle Links finden Sie auch online unter: de.elv.com/elvjournal-links

Material	Artikel-Nr.
JOY-iT Entwicklungsplatine NodeMCU mit ESP32	145164
ELV Steckplatine/Breadboard mit 830 Kontakten, schwarze ELV-Version	250986
ELV Bausatz Prototypenadapter für Steckboards PAD1	153761
ELV Bausatz Prototypenadapter für Steckboards PAD2, linear	154712
ELV Bausatz Prototypenadapter für Steckboards PAD3, passiv	154743
ELV Bausatz Prototypenadapter für Steckboards PAD4, digital	155107