I. Experimente mit dem micro:bit

1. Was ist der micro:bit?

Mikrocontroller sind **Einplatinencomputer**. Unser BBC micro:bit ist zwar kein kleiner PC – ihm fehlt ein eigenes Betriebssystem – aber wir können Daten ein- und ausgeben, zum Beispiel über Tasten oder Leuchtdioden. Wir können allerlei externe Elektronik an seine Pins anschließen. Und vor allem: Wir können ihn programmieren. Dabei bringt er bereits einige **Sensoren und Aktoren** direkt **„on board"** mit, d. h. wir müssen sie nicht selbst noch anschließen: Eine LED-Matrix zur visuellen Ausgabe, zwei Eingabeknöpfe, einen Beschleunigungs- und Lagesensor, einen Kompass sowie diverse Funkmodule zur kabellosen Kommunikation.

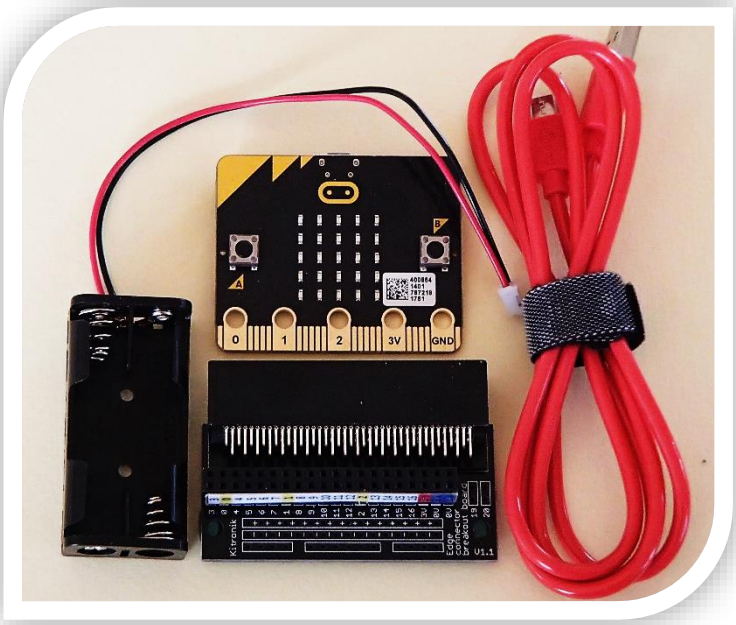


Abbildung 1: micro:bit mit mobiler Stromversorgung, USB-Kabel und Platinenstecker

Vertiefende Recherche: *Embedded Systems*

Mikrokontroller finden sich in etlichen Geräten des Alltags wieder! Recherchiere und notiere deine Ergebnisse knapp hier!

_____

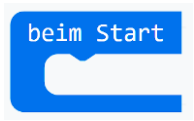
_____

_____

2. Programme schreiben und starten

Eine von vielen Entwicklungsumgebungen (auch **IDE** = *integrated development environment*) für den micro:bit ist die kostenfreie, blockbasierte **Makecode**-IDE von Microsoft, die unter folgendem Link erreichbar ist: https://makecode.microbit.org/. Ein Installieren ist nicht nötig, sie läuft direkt im Browser.

Die verschiedenen Anweisungen gliedern sich in mehrere Kategorien. Wir ziehen sie mit der Maus ins Programmierfenster. Aus der Kategorie **Grundlagen** sind zwei besonders wichtig:

Kategorie **Grundlagen**	
	Eine Sequenz, eine Abfolge von Anweisungen, innerhalb der blauen Klammer wird **einmalig beim Start** des Mikrokontrollers ausgeführt. Man nennt diesen Vorgang auch **Initialisieren**. **Tipp**: Soll der micro:bit beim Start nichts unternehmen, so lasse die Klammer leer. Füge Sie aber auf jeden Fall in Dein Programm ein!
	Die Sequenz innerhalb der blauen Klammer wird nach der obigen Start-Anweisung **immer wieder** ausgeführt. Es handelt sich dabei um eine **Endlosschleife**.

Arbeitsauftrag

1. Schreibe selbst ein eigenes Programm für den micro:bit! Verwende beide obigen Befehle.

Tipp: Ein möglicher Ausgangspunkt ist dabei rechts abgebildete Programm.

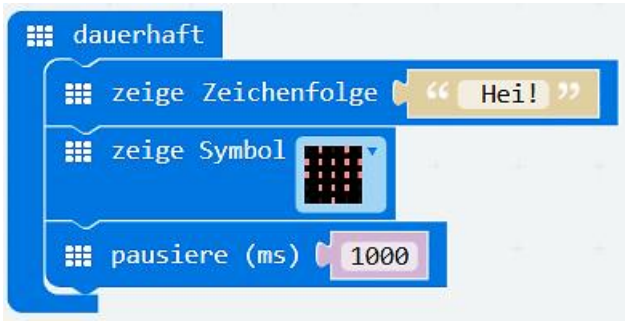


Abbildung 2: Ein erstes Programmbeispiel für den micro:bit

So lädst du das Programm auf den micro:bit

1. Du lädst die hex-Datei mittels Klick auf die Herunterladen-Schaltfläche (links unten) vom Browser in deinen Downloadordner herunter.

2. Du kopierst die hex-Datei auf den micro:bit; nach kurzem Blinken startet er das Programm.

Tipp: Du kannst hex-Dateien auch wieder **öffnen**! Dazu genügt links oben im Hauptbildschirm ein Klick auf: ***Projekte → Datei importieren***: Auswählen der hex-Datei

Arbeitsauftrag

2. Probiere auch die anderen Befehle aus der Kategorie **Grundlagen** aus.

a) Was passiert, wenn man den Befehl „pausiere“ (s. oben) aus dem Programm entfernt?

_____

b) Klicke in die 5x5 Kästchen bei „zeige LEDs“. Was passiert?

_____

c) Ersetze das Argument („Hei!“) der Anweisung **zeige Zeichenfolge** im obigen Programm durch ein einziges Zeichen Wiederhole Teilaufgabe a) und schildere deine Beobachtung! Probiere es danach mit dem Anzeigen einer Ziffer. Gibt es Unterschiede?

_____

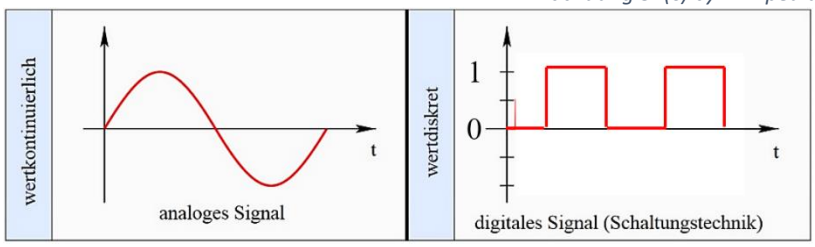
3. Digitale Eingaben und Pins

In der zweiten Kategorie (von oben gezählt), **Eingabe**, gibt es verschiedene Möglichkeiten, wie der micro:bit seine Umgebung wahrnimmt bzw. durch diese gesteuert werden kann.

Ein wichtiger Bestandteil sind dabei die **Pins** des micro:bit: Sie sind die messingfarbene Leiste an der unteren Kante des Mikrokontrollers, wobei fünf von ihnen größer sind und einen Namen besitzen.

Vertiefung: *Digital vs. analog*

Abbildung 3: (c) by Wikipedia



Was ist der Unterschied zwischen analogen und digitalen Signalen? Worin besteht der Vorteil digitaler Signale?

Recherchiere!

Arbeitsaufträge

0. Wikipedia schreibt über **Digitaltechnik** folgendes:

> Die in der Praxis bedeutsamste Form stellt die **binäre Digitaltechnik** dar, die nur **zwei diskrete Signalzustände** umfasst. Diese werden üblicherweise als **logisch null (0) und als logisch eins (1)** bezeichnet.

Formuliere *knapp* und in eigenen Worten, was ein (**binäres) digitales Signal** auszeichnet.

_____

_____

_____

1. Teste nebenstehendes Programm auf dem micro:bit!

Tipp: Um eine Eingabe auf dem Pin zu registrieren, musst du mit Hilfe einer Krokodilklemme den jeweiligen Pin mit der Erdung (≙ ground ≙ GND ≙ 0 Volt) verbinden.

2. Schaffst du es statt der Nummer die **Temperatur** auszugeben?

3. Verlängere die Pausieren-Anweisung in der dauerhaft-Schleife und betätige erneut Knopf A und Pin P0.

a) Beschreibe, was passiert.

_____

b) Wo wird folglich die Anzeigedauer der **Eingabe**-Schleifen gesteuert?

_____

Abbildung 4: Digitale Eingaben am micro:bit mittels Pin und Taste.

4.*-Aufgabe für Schnelle:

Leider verhält sich der micro:bit sowohl beim Drücken von Knöpfen auch als beim Betätigen der Pins nicht ganz so intuitiv wie gewünscht – selbst wenn die Erkenntnisse aus Aufgabe 3 beachtet werden!

Probiere aus, was bei einem langen Betätigen der Taste bzw. des Pins passiert. Achte darauf, wann (beim Drücken oder beim Loslassen) der micro:bit die Eingabe registriert! Notiere deine Ergebnisse untenstehend.

Knopf A	**beim ersten Drücken**	**beim Loslassen**
lange gedrückt		
kurz gedrückt		

Pin P0	**beim ersten Kontakt**	**beim Loslassen**
langer Kontakt		
kurzer Kontakt		

Vertiefung: Das obige Verhalten des micro:bit nennt man auch ***Flankensteuerung***. Eine steigende bzw. fallende Flanke entspricht dem Anstieg bzw. Abfall der anliegenden Spannung.

II. Miniprojekt: Alarmanalage

1. Basis-Projekt

Einen Teil der nötigen Befehle kennen wir schon aus der Kategorie **Eingabe** – jetzt werden noch Befehle der Kategorie **Logik** gebraucht: Wichtig ist besonders der **wenn-dann-ansonsten**-Befehl, den du über einen Klick auf das Zahnradsymbol links oben anpassen kannst.

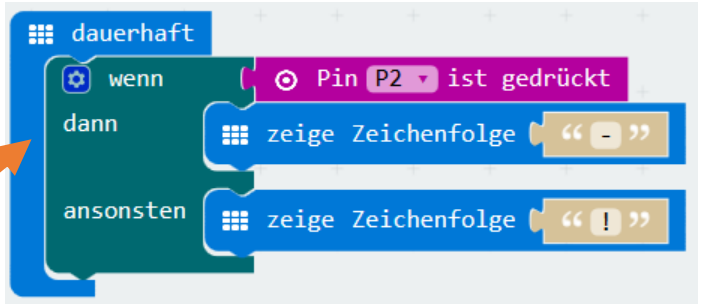


Abbildung 5: Codeschnipsel einer einfachen Alarmanlage.

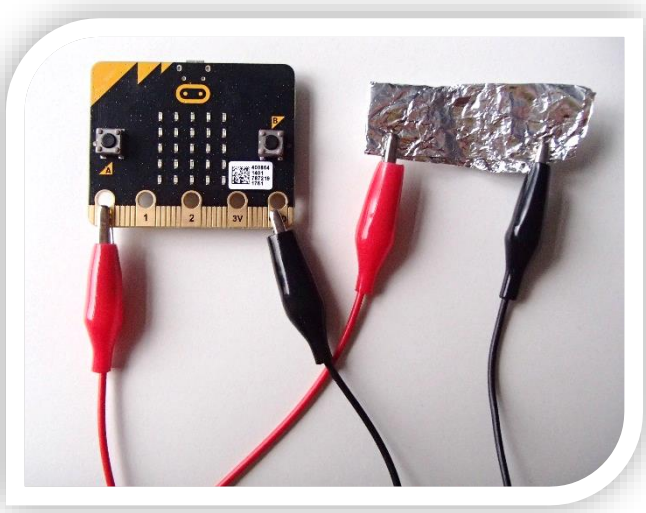


Abbildung 6: Alarmanlage mit Schokopapier.

Projektidee
Bau dir eine Alarmanlage, die du im Klassenraum installierst und die einen realen Gegenstand (Fenster, Wasserglas etc.) sichert!

Tipp: An die fünf großen Pins kannst du mit Krokodilklemmen Elektronik anschließen. Mit der Alufolie könntest du zum Beispiel eine Türe, Brotdose o. Ä. versiegeln.

Cleverer Dieb als Problem?!

Was passiert bei deiner Alarmanlage, wenn ein Dieb das Schokoladenpapier wieder zusammenklebt? Warum ist das so? Notiere.

_____

_____

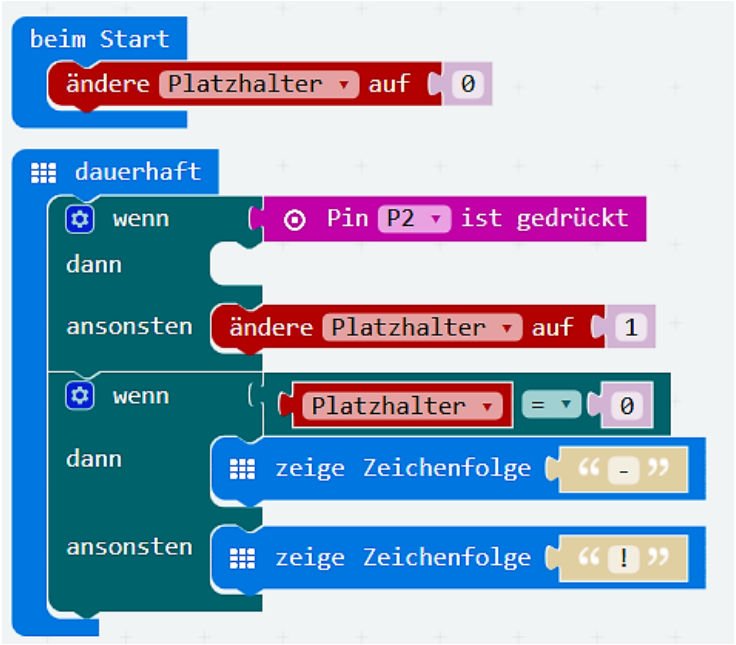
_____

_____

2. Ausbaumöglichkeiten der Alarmanlage

Projektwahlmöglichkeit 1: Variablen zum Speichern von Zuständen

Schwierigkeitsgrad: mittel bis schwer



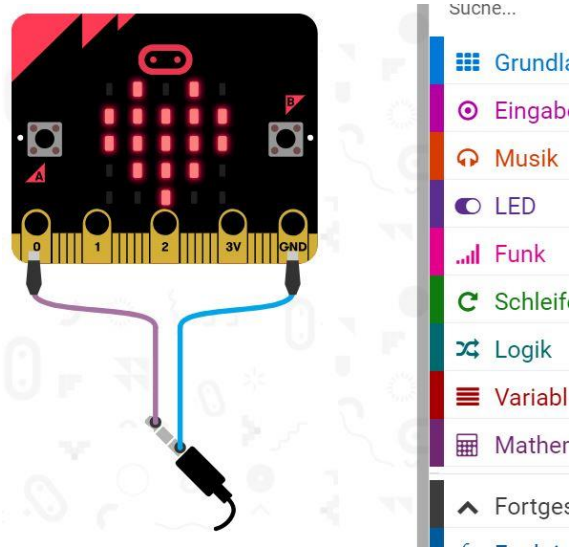
Idee, das Diebstahl-Problem zu lösen, ist eine **Variable** anzulegen, die sich „merkt“, ob der Stromfluss zum Pin jemals unterbrochen wurde, oder stets intakt war.

Tipp: Du kannst dich am nebenstehenden Programm orientieren.

Abbildung 7: Beispiel für Alarmanlage mit Variable.

Projektwahlmöglichkeit 2: Audioausgabe am micro:bit

Schwierigkeitsgrad: leicht bis mittel



Mithilfe der Befehle der Kategorie **Musik** kannst du ein Audio-Ausgabegerät (Kopfhörer, Summer, Lautsprecher etc.) an den micro:bit anschließen.

Tipp: Der Makecode-Simulator zeigt dir, wie du den Klinkenstecker anschließen kannst.

Abbildung 8: Makecode-Simulator nach Einfügen eines „Musik"-Befehls erklärt Anschluss eines Klinkensteckers.

3. Kabellose Datenübertragung

Wir wollen unser Alarmanlagenprojekt fortsetzen und diesmal den Alarm – egal, ob visuell oder akustisch – nun an einem wenige Meter entferntem Ort, d. h. auf einem **zweiten micro:bit**, wiedergeben. Die Befehle der Kategorie **Funk** helfen uns dabei.

Der Übersichtlichkeit halber nennen wir den **sendenden micro:bit** in am zu sichernden Objekt ***S***(ender), den **empfangenden micro:bit** in deiner Hand ***E***(mpfänger)**.**

Aufgabenstellungen

1. Teste untenstehendes Programm auf den beiden micro:bits.

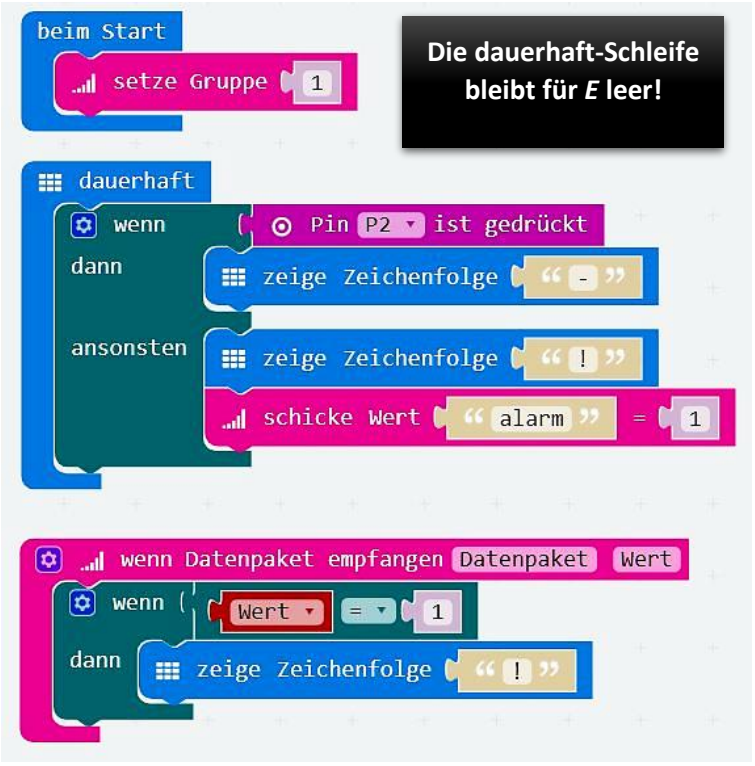


Abbildung 9: Funkalarmanlage mit dem micro:bit.

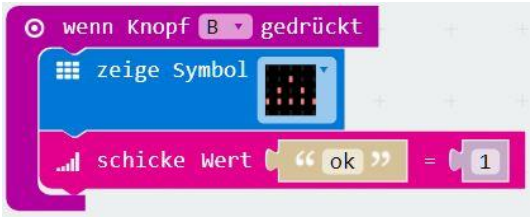
Wichtig: Verwende eine andere Gruppennummer! Denn diese legt den Funkkanal fest.

Vertiefung: ***Kanäle beim WLAN***

Wenn du Probleme mit deinem WLAN zu Hause hast, so kann es sein, dass das Umstellen des Funkkanals helfen kann. Recherchiere dazu!

2. Die Alarmanlage soll eine Testfunktion erhalten, damit überprüft werden kann, ob die Funkverbindung funktioniert (*S* und *E* also z. B. nicht zu weit voneinander entfernt sind). Dazu soll beim Drücken des Knopfes B an *S* ein Signal an *E* gesendet werden, der dies dann ausgibt.

Wird an *S* der Knopf B gedrückt, so soll er folgendes an *E* senden:



a) Ergänze zunächst nur diese Sequenz in der Alarmanlage. Was passiert beim Drücken von B?

_____

_____

b) Es ist also nötig, dass verschiedene Datenpakete (anhand ihres Namens) unterschieden werden können. Setze dies um.

Tipp: Nebenstehender Code-Auszug kann dir dabei helfen.

Vertiefung: ***Protokolle***

Die Art und Weise, wie unsere beiden micro:bits miteinander kommunizieren, d. h. insbesondere der Funkkanal, die Bezeichnungen der versendeten Datenpakete und die Bedeutung derer Werte, müssen wir selbst festlegen. Die Vorschrift, wie dies abzulaufen hat, nennt man **Netzwerk-Protokoll**. Informiere dich über Protokolle, die du tagtäglich im Internet verwendest. Notiere sie hier!

_____

_____

_____

c) *-Aufgabe für Schnelle: Ergänze die Alarmanlage um weitere sinnvolle Funkfunktionalitäten, wie z. B. das Senden der **Lichtstärke** von Sender zu Empfänger. Wie könnte die Lichtstärke beim Entdecken von Einbrüchen helfen?

_____

_____

Implementiere anschließend eine geeignete Lösung.

d)*-Aufgabe für Experten: micro:bit *E* soll ohne Reset wieder in den Grundzustand übergehen. Beachte: Es ist dann eine zusätzliche Lösung (mit Variablen) nötig, damit E einen cleveren Dieb, der Pin P2 wieder verbindet, nicht „vergisst“.

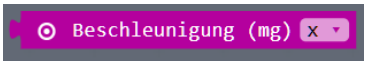
III. Endprojekt: Farbmischer & Co. mit dem micro:bit

1. Basisausbau: Lagesensor des micro:bits als Steuerungsgerät

benötigte Hardware in dieser Lektion
2 micro:bits
3 LEDs (rot, blau, grün)
1 Platinenstecker

Zunächst gilt es nur die Lageposition des micro:bit zu erfassen und auf seiner LED-Matrix auszugeben. Folgende Anweisung ist dabei hilfreich:

Kategorie **Eingabe**

	Gibt die Beschleunigung es micro:bit als Zahl pro Komponente (x-, y-, z-Richtung) aus.

Arbeitsaufträge

1. Gib die **Beschleunigung** (zunächst der x-Komponente) auf der LED-Matrix des micro:bits aus. **Tipp:** Dazu genügen zwei Befehle in der **dauerhaft**-Schleife.
2. → ***Wettbewerb***: Wer erhält den größten Wert?! 
3. Leider entspricht die Ausgabe aus Teilaufgabe 1 nicht unbedingt unseren Vorstellungen: Sie ist a) recht langsam und b) scheint etwas an der ausgegebenen Zahl für die z-Komponente komisch.

    1. Was ist der größtmögliche und kleinstmögliche Wert der ausgegebenen Zahl? Überlege dir, wie sie kompakter („gerundet“) ausgegeben werden kann. Notiere deine Antwort.

        Tipp: Die Aufgabe ist ähnlich zum Umrechnen in größere Einheiten.

        Tipp für Experten: Überlege Dir die Umrechnungszahl anhand des Wertebereichs.

        _____

        _____

        _____

2. Der ausgegebene Wert für die z-Komponente hat ein negatives Vorzeichen. Wieso könnte dies so sein?

   Ändere dies ab!

   _____

3. *-Aufgabe für Experten: Bisher konnte man immer nur eine der drei Komponenten ausgeben; ändere dies so ab, dass man per Taster (A, B, A und B) umschalten kann – ohne das Programm erneut auf den micro:bit laden zu müssen!

Zum Beachten beim Farbmischen:

- Schau nicht die LEDs an, sondern eine Fläche, die die Farben streut!
  Noch besser wird der Effekt mit zwei (nicht direkt aufeinanderliegenden) Papierlagen:

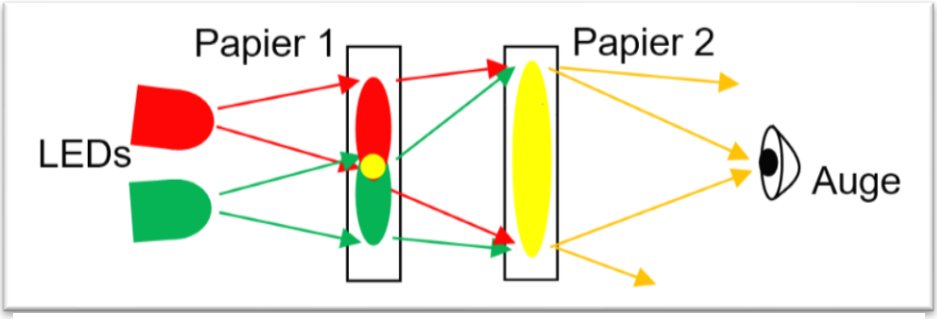
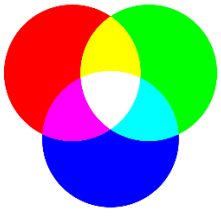


Abbildung 10: Schema der Streuung von roter und grüner LED auf Papierlagen.

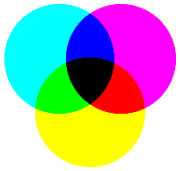
- Mit unserem Farbmischer kann man nicht (!) alle Farben mischen – dies liegt an der Art und Weise, wie wir ihn ansteuern. Finde selbst ein Beispiel dafür und notiere!

  _____

Vertiefung: ***Additive vs. subtraktive Farbmischung***



(Flach-)Bildschirme, Handydisplays und viele weitere Anzeigen arbeiten auf Basis von additiver Farbmischung (RGB) – so auch unser Farbmischer. Drucker hingegen arbeiten auf Basis subtraktiver Farbmischung (CYM(K)). Recherchiere die Unterschiede und ordne die beiden Graphiken richtig zu.



_____

_____

_____

2. Basisausbau: LEDs per Lagesensor steuern

Schließe nun den Platinenstecker an den micro:bit an und verbinde es mit den drei LEDs. Wir wollen diese nun per Beschleunigung steuern. Schreibe dazu (wie in Kapitel I.3) auf die Pins – diesmal aber **analog**! Sie werden mit **Werten von 0 bis 1023** angesteuert.

Abbildung 11: micro:bit auf Platinenstecker mit drei LEDs.

Arbeitsaufträge:

1. Ergibt es Sinn negative Werte auf die Pins zu schreiben? Prüfe nach, was passiert, wenn man dies tut.

   _____

   _____

2. * für Experten: Der Beschleunigungssensor umfasst einen Wertebereich von -1023 bis 1023, die analogen Pins aber nur von 0 bis 1023. Fällt Dir eine Lösung ein, wie man die LEDs „besser“ steuern könnte? Formuliere zunächst eine Formel!

   _____

   _____

3. als Projektoption: Entwerfe einen Farbmischer, der per Funk gesteuert wird. Untenstehend dazu ein möglicher Code für das Empfangsmodul.

Tipp: Auf Sender und Empfänger kann das gleiche Programm laufen!

3. individuelle Ausbaumöglichkeiten

Colour-Game

- Versuche möglichst schnell **verschiedene Farben** abzumischen!
  - besonders interessant sind zunächst die drei Grundfarben **Rot-Grün-Blau**
  - versuche auch ein möglichst passendes Weiß abzumischen
- Finden sich mehrere Teams für diese Idee, so wird ein **Klassenwettbewerb** veranstaltet, wer am schnellsten zufällig vorgegebene Farben abmischen kann!

Ausbauoptionen:

- **leicht**: Du kannst mithilfe von **Funk** auch mehr als einen micro:bit als Empfänger ansteuern. So ist es möglich, dass der **zweite Empfänger** z. B. die x-Komponente statt für blau für grün interpretiert.
  - Durch zwei verschiedene Farben hast Du einen Vorteil im **Wettbewerb**!
- **schwer**: Mithilfe von **Variablen** kann man Werte, z. B. per **Knopfdruck** speichern.

Eigene Stimmungsleuchte

- Viele Menschen mögen besonders beim Einschlafen Lampen, die angenehme Farbverläufe vollführen – wir wollen nun so eine bauen!

Ausbauoptionen:

- **leicht**: Du brauchst dabei nicht auf eine Lampe beschränkt zu sein, sondern kannst mithilfe mehrerer micro:bit, viele LEDs ansteuern (siehe ersten Tipp oben).
- **leicht**: Zwar können wir mithilfe von Rot-Grün-Blau viele Farben mischen, aber du kannst noch weitere LEDs am Klinkenstecker montieren.
- **mittel**: Mische dir mit dem bestehenden Farbmischer eine schöne Farbe ab und lasse dabei jeweils die Beschleunigungswerte für x-, y- und z-Richtung ausgeben. Notiere diese. Du kannst deine Notizen dazu verwenden, auch ohne Steuerung die Farbe dauerhaft ausgeben zu lassen. Per Knopfdruck könntest du **zwischen verschiedenen schönen Farben wechseln** lassen.
- **mittel**: Mithilfe der **pausiere**-Anweisung und dem An- und Ausschalten von Pins kannst du interessante **Blinkmuster** erstellen oder gar **morsen**!
- **mittel**: Mithilfe des integrierten Temperatursensors kannst Du beim Über-/Unterschreiten einer voreingestellten Temperatur eine Art „Alarm“ mithilfe der blauen bzw. roten LED ausgeben.
  → **schwieriger**: Kontinuierliche Farbübergänge von rot nach blau!
- **schwer**: Mithilfe von **Variablen** kann man Werte, z. B. per **Knopfdruck** speichern.
- außerdem: Du kannst noch **Soundeffekte** (vgl. S. 6) zur Stimmungsleuchte oder eine **Stoppuhrfunktion** hinzufügen – oder ein ganzes **Beleuchtungssystem** entwickeln! Du kommst bestimmt auf noch mehr Ideen!