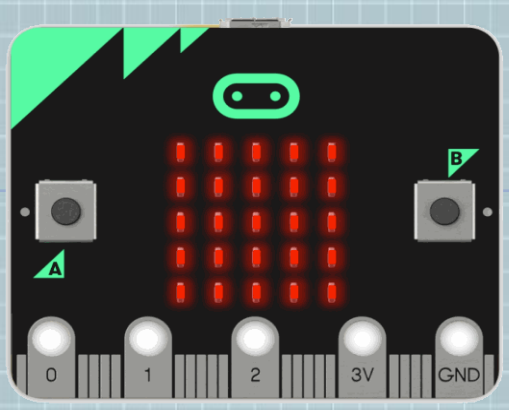
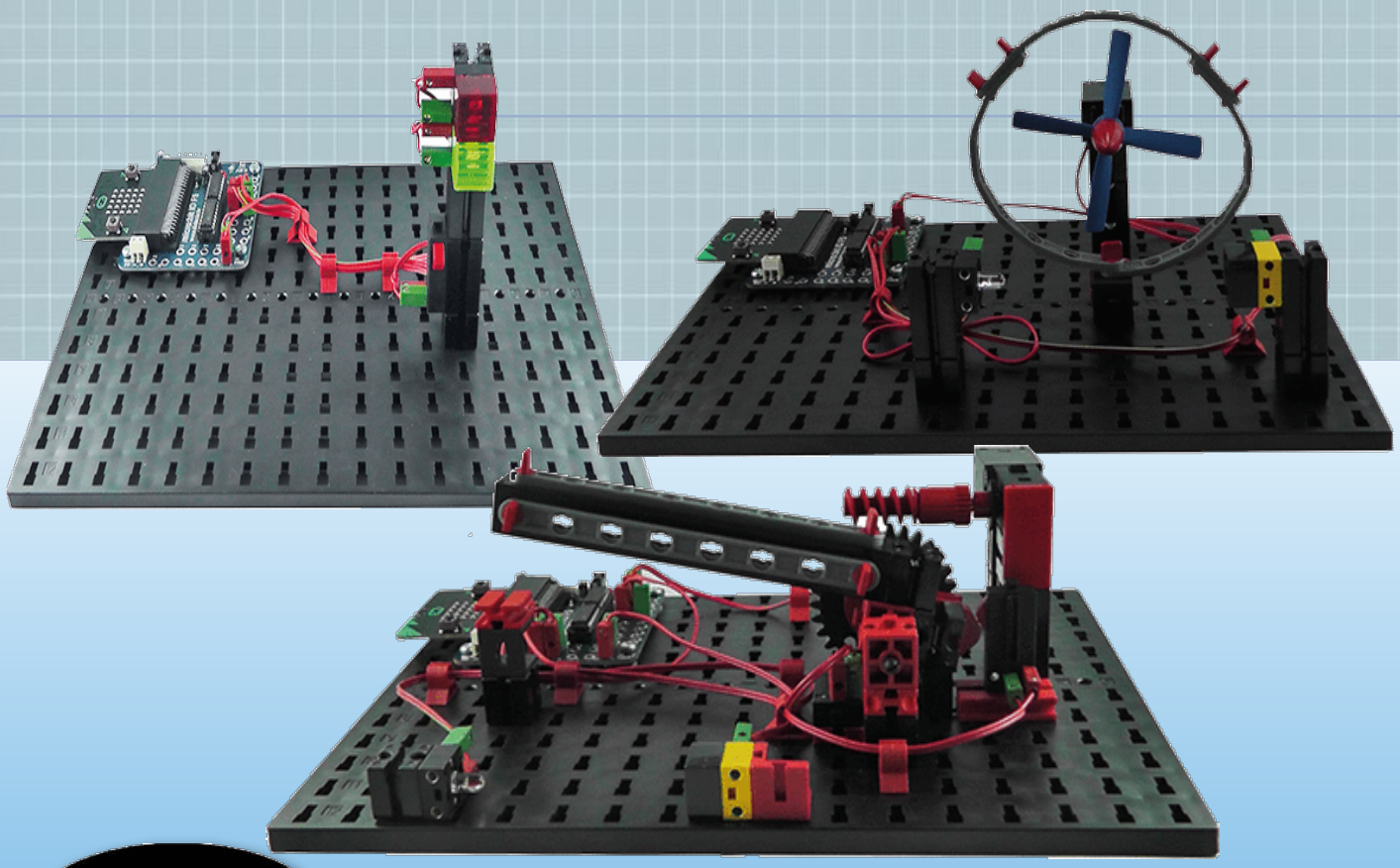
Programmieren in der Grundschule

fischertechnik

micro:bit





Microsoft

3 Models

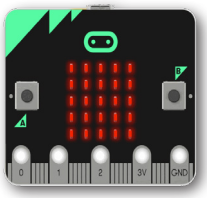
Inhalt

Vorwort

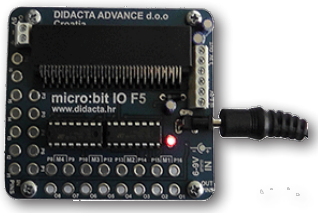
Hallo!
Ich darf mich kurz vorstellen -
mein Name ist RoBo und ich werde
dich bei der Erarbeitung der einzelnen
Aufgaben begleiten.

Bevor du mit den Aufgaben beginnst, muss ich dir noch ein paar wichtige Grundlagen erklären.

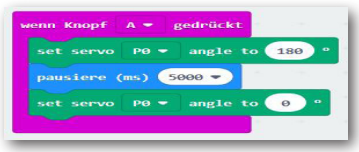
Mit den Bausteinen von fischertechnik baust du die entsprechenden Modelle auf, die du steuern möchtest.



Der micro:bit ist ein kleiner programmierbarer Computer. Dieser besitzt einen LED-Bildschirm, zwei Tasten, einen Lichtsensor, eine Bluetooth-Antenne, sowie einen Beschleunigungssensor und einen Kompass.



Am micro:bit IO F5 wird der micro:bit über eine Messerleiste angeschlossen. Alle Ein- und Ausgänge sind über Steckbuchsen erreichbar. Gleichzeitig kann eine externe Stromquelle angeschlossen werden.



Mit wenigen Klicks erstellst du mit makecode im Nu deine eigenen Programme für den Mikroprozessor und bringst die Modelle in Bewegung.

Viel Vergnügen bei der Durchführung der einzelnen Aufgaben.

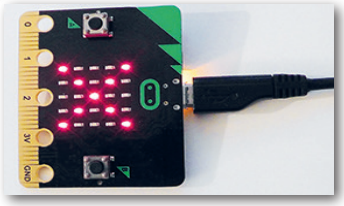
Dein RoBo

Vorbereitung

Bevor du mit dem Aufbau der Modelle und dem Programmieren beginnst, musst du einige Vorbereitungen treffen.

Packe zuerst den mikro:bit aus. Dieser wird in einem kleinen Karton verpackt geliefert.

Wichtig: Wenn du ihn später in die Steckerleiste des Adapters an deinem Modell einsetzt, müssen die Taster A und B nach oben zeigen.



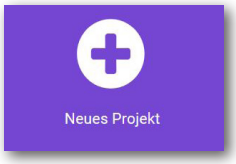
Schließe über ein USB-Kabel den mikro:bit an einem freien USB-Platz an deinem Computer an. Auf der Unterseite des micro:bit leuchtet eine gelbe Leuchtdiode. Die LED-Matrix auf der Oberseite zeigt ein rotes X.

Starte deinen Internetbrowser und öffne über

https://makecode.microbit.org

den Programmeditor.

Funktionstest



Klicke auf „Neues Projekt“. Es erscheint der Arbeitsbildschirm von makecode. Diesen werde ich dir im nächsten Kapitel noch genau erklären. Hier geht es nur darum, das Programm in Verbindung mit dem mikro:bit zu testen.

Stelle über „... mehr“ die Programmiersprache auf „Deutsch“ um.

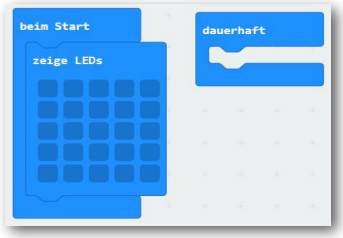
Klicke mit der Maus auf die Auswahl „Grundlagen“. Es erscheinen in diesem Block alle Befehle aus diesem Blockbereich.

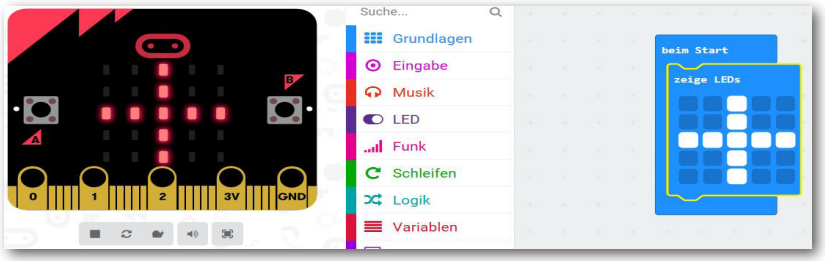
Für unseren Test verwendest du den Befehl „zeige LEDs“.

Ziehe diesen mit der Maus in die Leerfläche des Befehls „beim Start“.

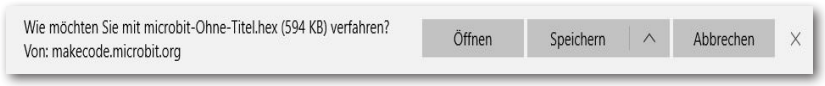
Wenn du auf die einzelen LEDs klickst werden diese aktiviert und in der Simulation dargestellt.

Stelle einmal ein Pluszeichen mit den LED´s dar. Dein Arbeitsbildschirm sollte wie das Bild zeigt, dargestellt werden.

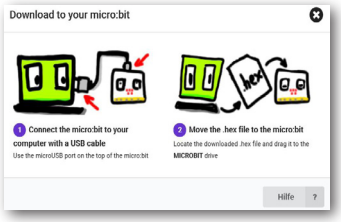




Mit der Schaltfläche „Herunterladen“ wird das Programm zum mikro:bit gesendet. Aktiviere die Schaltfläche und es erscheint am unteren Bildschirmrand folgende Zeile:



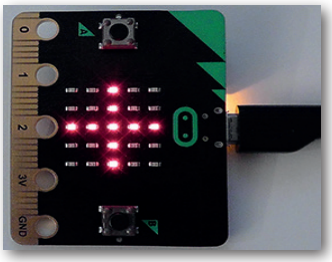
Aktiviere den Pfeil hinter „Speichern“ und wähle „Speichern unter“ aus.



Es erscheint der Arbeitsbildschirm „Speichern unter“. Wähle hier dein USB-Laufwerk aus, an dem du den micro:bit angeschlossen hast. Dieser wird optisch angezeigt.

Wichtig: Der Dateityp muss immer eine HEX-Datei sein.

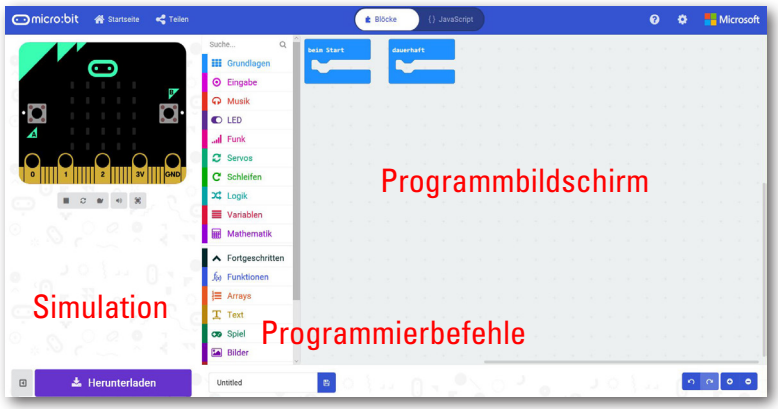
Wenn du auf „Speichern“ klickst, verlöschen die LED´s auf dem mikro:bit und die gelbe LED auf der Unterseite blinkt. Dies zeigt an, dass das Programm übertragen wird. Ist dies geschehen, wird das Pluszeichen angezeigt.



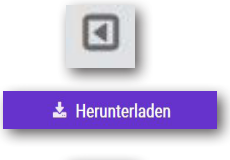
Wenn alles funktioniert hat, möchte ich dir den Editor etwas genauer erklären.

Editor

Der Bildschirm gliedert sich in drei Bereiche: Simulation, Programmierbefehle und den Programmbildschirm.



Weitere wichtige Schaltflächen:



Ein- und Ausschalten der Simulation

Herunterladen des Programms zum mikro:bit

Speichern des Projektes

Rückgängig machen

Vergrößern/Verkleinern

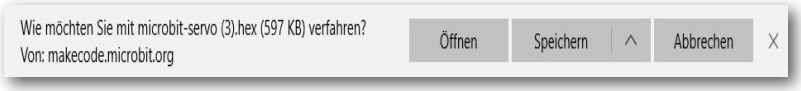
Simulation stoppen | Simulation neu starten | Zeitlupe

Ton stumm-schalten

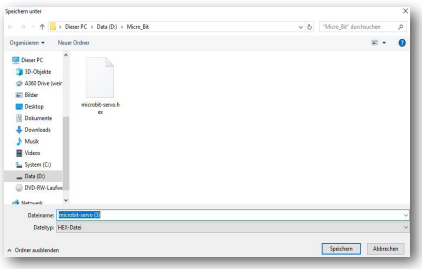
Vollbild starten

Speichern

Hast du ein Projekt fertiggestellt, musst du es speichern. Dazu wählst du die Schaltfläche „Speichern“ aus.



Klicke auf den Pfeil neben „Speichern“ und dort auf „Speichern unter“.



Wähle einen Ordner aus oder erzeuge einen neuen Ordnen, in den du dein Projekt speichern möchtest.

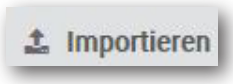
Wichtig: Vergebe einen aussagekräftigen Namen für dein Projekt.

Laden

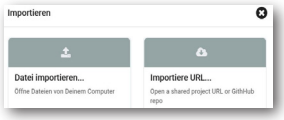
Möchtest du ein fertiges Projekt oder an einem Projekt weiterarbeiten, musst du es von deinem Speichermedium laden. Dazu benötigst du die Startseite. Wenn du makecode neu startest, befindest du dich in dieser.

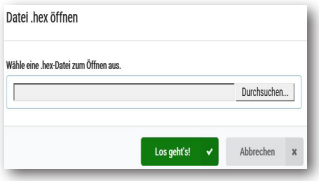
Deine Projekte findest du unter „Meine Projekte“.

Du kannst aber auch die Schaltfläche „Importieren“ auswählen. Es erscheint ein Kontextmenü. Wähle hier „Datei importieren“ aus.



Es erscheint ein weiteres Kontextfenster. Über die Schaltfläche „Durchsuchen“ gelangst du auf deinen Datenträger. Hast du dein Projekt gefunden, betätigst du „Los geht´s!.





Blöcke Befehle

In der Anleitung findest du den Begriff „Block“ und „Befehle“. makecode ist so aufgebaut, dass alle Befehle, funktionssortiert in Blöcken untergebracht sind. Ein Beispiel siehst du links für den Block „LED“.

Ok, das wäre auch geschafft und du kannst dich jetzt an das erste Modell wagen.

Ampel

Ampelanlagen kennst du sicherlich in verschiedenen Ausführungen. Fußgängerampeln oder ganzen Kreuzungsanlagen begegnest du fast täglich, so dass das Prinzip für dich nichts Neues mehr ist. Einfach ausgedrückt, Lampen werden in einer bestimmten Reihenfolge ein- und ausgeschaltet.

Was deine Ampel später ausmacht ist, dass sie über einen Taster, der sich am Ampelmast befindet, eingeschaltet wird.

Der Techniker schaltet zunächst in einem Schaltkasten die Anlage ein. Danach leuchtet die rote Lampe. Erst wenn der Fußgänger auf den Anforderungstaster drückt, schaltet die Anlage auf grün um und er kann die Straße überqueren.

Wenn du auf dem Heimweg von der Schule oder bei einem Spaziergang mit deinen Eltern an einer Fußgängerampel verbeikommst, dann beobachte diese einmal.

Stelle dir dabei folgende Fragen:

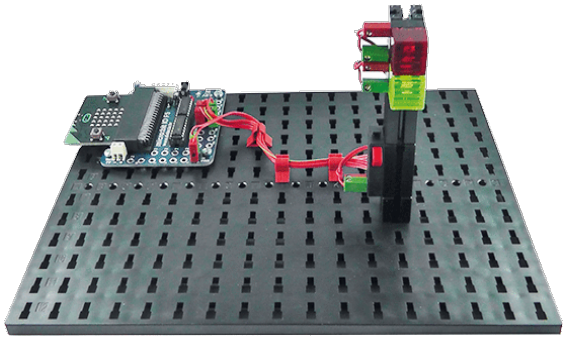
- Ist die Anlage eingeschaltet?
- Welche Lampe leuchtet?
- Besitzt die Anlage einen Signalanforderungstaster?
- Was geschieht wenn dieser gedrückt wird?
- Wie lange dauert die Grünphase?
- Was geschieht nach der Grünphase?

Notiere dir einfach die Lösungen auf einem Blatt Papier und verwende die Ergebnisse später bei den einzelnen Programmbefehlen.

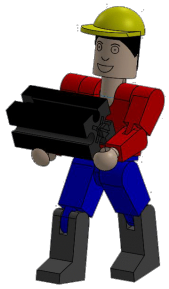
Wichtig: Bevor du die realen Modelle in Betrieb nimmst, kannst du deine Programme natürlich auch zuerst mit dem Programmsimulator testen.

Technische Anmerkung: Für alle Modelle benötigst du ein fischertechnik-Netzteil. Dieses ist nicht im Lieferumfang enthalten und muss von fischertechnik bezogen werden.

Fußgängerampel mit Anforderungstaster



Baue anhand der Bauanleitung dein erstes Modell auf.

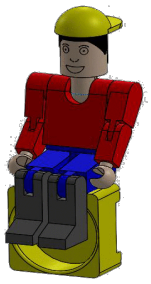


Verdrahte das Modell entsprechend des Schaltplans.

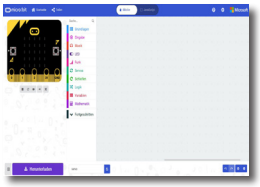
Für dieses Modell werde ich dir eine Aufgabe und eine Zusatzaufgabe stellen, die du dann in ein Steuerprogramm umsetzen musst.

ampel1.hex

Die Gesamtanlage soll durch einen Serviceschalter „Taster A“ am micro:bit eingeschaltet werden. Danach steht die Ampel auf rot. Erst wenn der Anforderungstaster am Ampelmast betätigt wird, schaltet Rot nach 2 Sekunden auf Grün um. Die Grünphase soll 5 Sekunden dauern. Danach schaltet die Anlage wieder auf Rot um und wartet auf eine erneute Betätigung des Anforderungstasters.

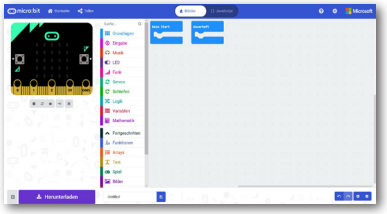


Das erste Projekt werden wir gemeinsam erstellen.



Starte als nächstes den Editor „makecode“. Es erscheint der dir schon bekannte Bildschirm.

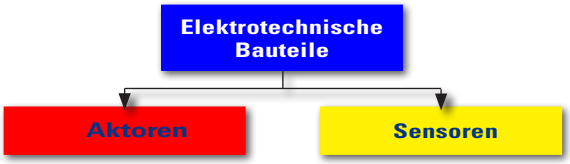
Damit du mit allen zur Verfügung stehenden Befehlen arbeiten kannst, erweiterst du mit „Fortgeschritten“ die Befehlsleiste.



Wichtig: Stelle sicher, dass du den „Fortgeschritten“ Bereich der makecode-Blöcke mit der entsprechenden Schaltfläche umgestellt hast.

Aktoren Sensoren

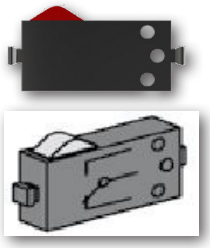
Halt, ich muss dir jetzt noch kurz die beiden elektrotechnischen Bauteile erklären, die in deinem Modell verbaut sind. Je nach Funktion nennt man sie „Aktoren“ und „Sensoren“.



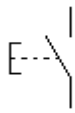
Aktoren heißen so, weil sie aktiv sind, sie tun etwas wie z.B. ein Motor oder eine Lampe. Mit Sensoren z.B. ein Taster, können Aktoren gesteuert werden.

Taster

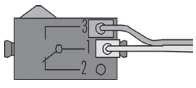
Beginnen möchte ich mit dem Sensor - Taster

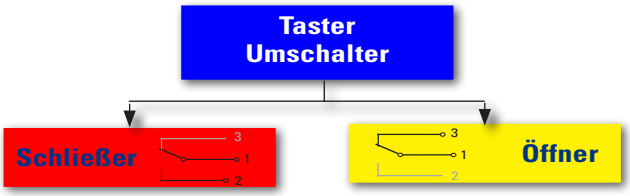


Taster zählen zu den Berührungssensoren. Betätigst du den roten Knopf, wird im Gehäuse ein Kontakt mechanisch umgelegt und es fließt Strom zwischen den Anschlüssen 1 und 3. Gleichzeitig wird der Kontakt zwischen den Anschlüssen 1 und 2 unterbrochen. Somit kannst du den Taster als Umschalter in deine Modelle einbauen.



Schaltzeichen





LED

Als Nächstes folgt ein Aktor, die Leuchtdiode oder LED

Der Name Leuchtdiode auch kurz LED genannt, kommt vom Englischen - light-emitting-diode. Übersetzt: ‚Licht-emittierende-Diode‘. Fließt durch die Diode elektrischer Strom (Durchlassrichtung), so strahlt sie Licht ab.

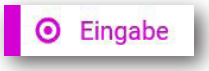
Im Baukasten findest du zwei LED-Bausteine. Du kannst sie einmal als normale Lampe aber auch später als Signalgeber bei einer Lichtschranke verwenden.


Schaltzeichen

Achte bei den elektrischen Verbindungen auf die richtige Polung.

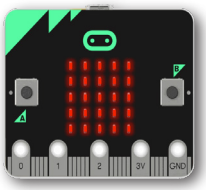
Die anderen Bauteile werde ich dir bei der jeweiligen Aufgabe erklären.

Jetzt ist alles Wichtige erklärt und du kannst an dein erstes Programm gehen.



Fangen wir mal der Reihe nach an. Jedes Programm, das du schreibst, beginnt immer mit einem „Start“-Befehl.

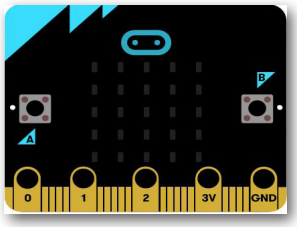
In der Aufgabe heißt es, dass das Programm durch Drücken der Taste A auf dem micro:bit gestartet wird.

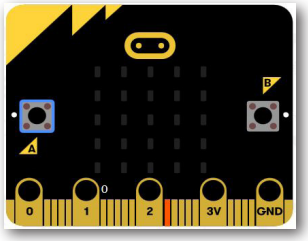


Du verwendest den Befehl „wenn Knopf ... gedrückt“ aus dem Block „Eingabe“. Ziehe den Befehl auf deinen Programmbildschirm.

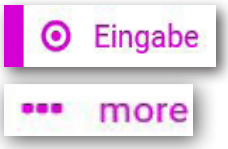
Wird der Knopf A am mikro:bit gedückt, soll die rote Lampe (LED am PIN13) eingeschaltet werden. Docke den Befehl aus dem Block „Pins“ in den Zwischenraum von „wenn Knopf --- gedrückt“ ein. Ändere den Eintrag P0 auf P13 und den Wert 0 auf 1 um.

Der erste Teil des Programms ist fertig. Du kannst dir das Ergebnis schon mal in der Simulation ansehen. Klicke mit der Maus auf den Taster A. Der Pin 1 wird auf ein geschalten, mit einer logischen 1 versehen und rot dargestellt. Hast du das Modell schon am Computer angeschlossen, müsste dort auch die rote Lampe leuchten.





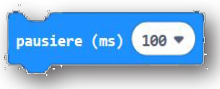
Die Blöcke „beim Start“ und „dauerhaft“ werden nicht benötigt und können gelöscht werden. Rechter Mausklick auf den Befehl und im Kontextmenü „Delete Blocks“ auswählen.



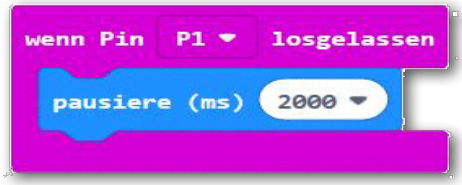
Als Nächstes erstellst du einen Programmteil, mit den Befehl „wenn Pin P1 losgelassen“aus dem Block „Eingabe“ eingebettet wird. Ziehe den Befehl auf deinen Arbeitsbildschirm.

Ändere den Eintrag „P0“ auf „P1“ um.

Es folgt jetzt ein Pausebefehl. Diesen findest du im Block „Grundlagen“ „pausieren (ms) xxxx“.

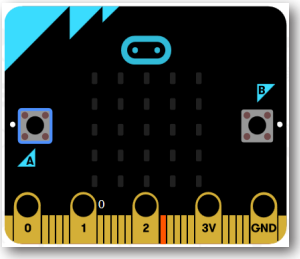


Ziehe den Befehl an die Andockstelle und ändere die Zeitangabe in 2000 für 2 Sekunden um.



Was geschieht anschließend? Wird die Taste am „P1“ gedrückt, soll die rote Lampe ausgehen und die grüne Lampe angehen. Die rote Lampe ist am „Pin13“ und die grüne Lampe am „Pin12“ angeschlossen. Hier benötigst du aus dem Block „Pins“ den Befehl „schreibe digitalen Wert ....“. Diesen dockst du unter den Pausebefehl an.

Da du den Befehl zweimal benötigst, kannst du ihn einfach duplizieren. Klicke mit der rechten Maustaste auf den Befehl und wähle im Kontextmenü „Duplizieren“ aus. Den zweiten Befehl dockst du unter den letzten Eintrag an. Ändere im ersten Befehl „P1“ auf „P13“ und schalte von „1“ auf 0“ um. Im zweiten Befehls änderst du von „P1“ auf „P12“ und von „0“ auf „1“ um.



Du kannst dein Teilprogramm wieder testen. Betätige die virtuelle Taste A für die Simulation. Pin 13 wird rot dargestellt.

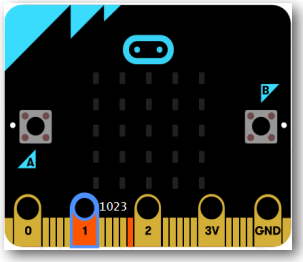
Sende das Teilprogramm zum micro:bit (siehe Seite 5)

Herunterladen

Betätige auch am mikro:bit die Taste A. Es leuchtet die rote Lampe.

In der Simulation klickst du auf den Pin1. Dieser wird als betätigt dargestellt.

Nach 2 Sekunden geht die rote Lampe aus wie auch in der Simulation P 13. Dafür schaltet die grüne Lampe oder Pin 12 ein.



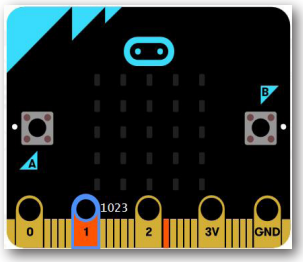
Drückst du am Ampelmast die Fußgängertaste, geht die rote Lampe aus und die grüne Lampe leuchtet.

In der Aufgabe heißt es weiter, dass nach 5 Sekunden die grüne Lampe ausgehen und die rote Lampe wieder leuchten soll.

Füge aus „Grundlagen“ den Befehl „pausiere (ms) xxxx“ gefolgt von 2 Befehlen aus „Pins„ - „schreibe digitalen Wert ...“ ein.

Die Pausezeit änderst du auf 5000 für 5 Sekunden um.

Ändere die Pinbelegung auf „P12“ und „P13“ um. Schalte „P13“ auf „1“ und „P12“ auf „0“.



Führe auch hier zuerst den virtuellen Test durch. Nach der Pausezeit von 5 Sekunden wird Pin12 ausgeschalten und Pin13 leuchtet wieder.

Sende das Gesamtprogramm zum micro:bit und starte es am Modell. Auch hier springt die grüne Lampe nach einer Pause von 5 Sekunden auf Rot um.

Fußgängerampel mit Blinklicht

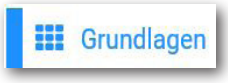
Kommen wir zum zweiten Teile der Aufgabe.

ampel2.hex

Nach Ablauf der 5 Sekunden soll die grüne Lampe dreimal blinken. Die Blinkfrequenz beträgt 1 Sekunde. Anschließend wird die Ampel auf rot umgeschaltet.

Hier verwendest du den Befehl „x-mal wiederholen“ aus dem Block „Schleifen“. Docke diesen nach der Pause von 5 Sekunden an.

Ändere die Wiederholung von „4“ auf „3“.

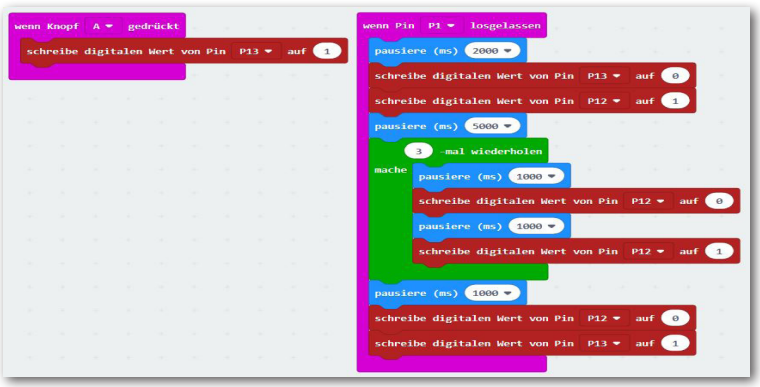


Füge eine Pause von 1 Sekunde ein.
Anschließend folgt der Befehl „ „schreibe digitalen Wert .... P12 auf 0“

Dann folgt ein weiterer Pausenbefehl von 1 Sekunde und der Befehl zum Einschalten des P12.

Füge nach dem Schleifenbefehl noch einen Pausenbefehl von 1 Sekunde ein.

Somit ist auch der zweite Aufgabenbereich programmiert. Speicher die Datei auf deinem Computer ab. Teste wie schon gewohnt einmal virtuell und das zweite Mal an deinem Modell.



Händetrockner

In deinem Bad Zuhause wirst du so etwas sicher nicht haben. Da hängt ein großes Handtuch am Haken. Aber in öffentlichen Toiletten, in der Toilette deiner Schule oder im WC-Bereich von Gaststätten sind an der Wand meist elektrische Gebläse montiert, die dir mit warmer Luft die Hände trocken pusten.

Eine gute Erfindung, besonders wenn du so ein modernes Gerät vorfindest, an dem du keinen Knopf betätigen musst um es einzuschalten. Einfach die Hände hinhalten und los geht's.

Ein Händetrockner mit berührungslosem Ein- und Ausschalter kannst du dir jetzt mithilfe der Bauanleitung aufbauen und nach Schaltplan verdrahten.

Stelle dir auch für diese Aufgabe folgende Fragen:

- Wie wird der Händetrockner eingeschaltet?
- Leuchtet eventuelle eine Kontrollanzeige?
- Wann schaltet der Händetrockner wieder ab?
- Wie lange dauert die Trockenzeit?

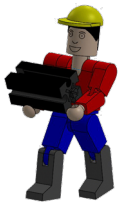
Notiere dir einfach die Lösungen auf einem Blatt Papier und verwende die Ergebnisse später bei den einzelnen Programmbefehlen.

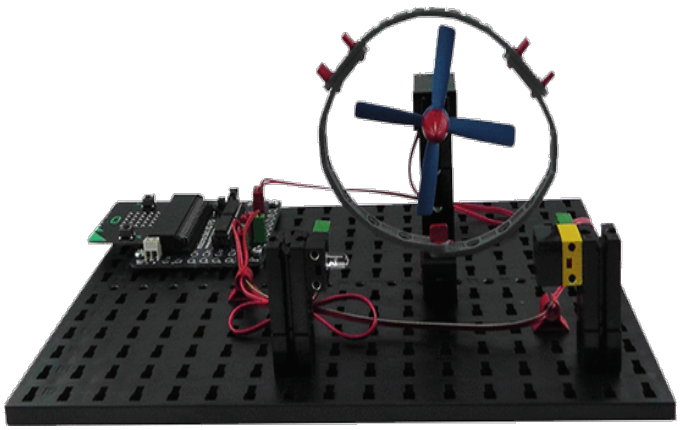
Viel Spaß beim Lösen dieser Aufgabe.

Dein RoBo

Händetrockner mit Lichtschranke

Baue anhand der Bauanleitung dein zweites Modell auf.



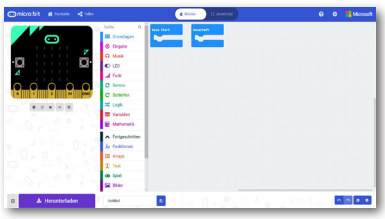


Verdrahte das Modell entsprechend des Schaltplanes.

Auch für dieses Modell werde ich dir eine Aufgabe und eine Zusatzaufgabe stellen, die du dann in ein Steuerprogramm umsetzen musst.

ventilator1.hex

Programmiere den Händetrockner so, dass wenn ein Benutzer in die Lichtschranke greift, der Motor mit dem Propeller startet. Nimmt er die Hand wieder aus der Lichtschanke heraus, muss der Motor stoppen.



Auch bei dieser Aufgabe werde ich dir natürlich helfen.

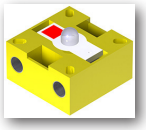
Starte wie bei der vorherigen Aufgabe deinen Editor.

Wichtig: Stelle auch hier wieder sicher, dass du den „Fortgeschrittenen“ Block mit der entsprechenden Schaltfläche umgestellt hast.

Bevor du mit dem Programmieren beginnst, muss ich dir noch einen weiteren Sensor und Aktor vorstellen.

Für die Lichtschranke, die du zum Ein- und Ausschalten des Lüftermotors benötigst, brauchst du die dir schon bekannte LED sowie einen Fototransistor.

Fototransistor

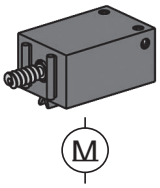


Ein Fototransistor ist ein elektronischer Schalter (Sensor), der auf Licht reagiert. Sicherlich hast du dich schon gefragt, wie im Kaufhaus die Eingangstür automatisch aufgeht, ohne dass du einen Taster oder Schalter betätigst.

Schaltzeichen

Hierfür wird eine Lichtschranke eingesetzt, die aus einer Lichtquelle (Sender) und einem Sensor (Empfänger) besteht. Im Baukasten wird ein LED-Baustein als Sender und ein Fototransistor-Baustein als Empfänger verwendet.

Motor



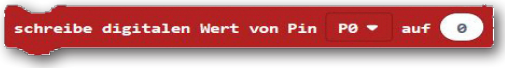
Schaltzeichen

Der Motor ist ein Gleichstrommotor (Aktor). Dieser wandelt elektrische Energie in mechanische Energie um. Dadurch entsteht eine Drehbewegung der Motorachse.

Den Motor kannst du direkt mit dem micro:bit steuern.

So, du hast dein Modell aufgebaut und verdrahtet - jetzt musst du es nur noch programmieren.

Du hast den Editor ja schon gestartet und beginnst mit dem Block „beim Start“. Dieser ist auf dem Bildschirm schon vergegeben, so dass du gleich einen weiteren Befehl einbauen kannst.



Wie du aus der Bauanleitung erkennen kannst, wird der Motor am „Pin16“ und „Pin15“ angeschlossen. Pin16 soll der Minuspol und der Pin15 der Pluspol werden. Deshalb wird der Pin16 beim Start auf „0“ gesetzt. Die LED ist am Pin8 angeschlossen. Da sie dauerhaft leuchten soll, kannst du den entsprechenden Befehl in den Programmteil „beim Start“ einbinden.

Für den weiteren Programmverlauf benötigst du „dauerhaft“. Auch dieser Befehl steht schon auf dem Programmbildschirm.

Da du mit einem Fototransistors arbeitest, musst du den Wert den dieser liefert von einem digitalen Wert auf einen analogen Wert umwandeln bzw. einlesen.

Digital Analog

Bevor du weiter arbeitest, muss ich dir zwei Begriffe erklären „Digital“ und „Analog“. Was versteht man darunter?

Digitale Werte können nur zwei Zustände haben 0 oder 1, ja oder nein. Ein Beispiel: Eine Lampe brennt (1) oder sie ist aus (0).

Benötigst du einen analogen Wert, kann dieser sich in einem bestimmten Bereich befinden und somit mehrere Werte haben. Z.B. wenn du einen bestimmten Temperaturbereich ermitteln möchtest - einschalten eines Lüftermotors bei einer Außentemperatur von 20 Grad bis 24 Grad.

Bau zuerst den Befehl „schreibe digitalen Wert ...“ ein. Schalte „P0“ auf „P1“ um, da du den Fototransistor an diesem Pin angeschlossen hast.

Unter dem Block „Pins“ findest du einen Befehl „analoge Werte von Pin ...“. Diesen platzierst du auf der „0“ des ersten Befehls.

Jetzt benötigst du einen Befehl, der abfragt „Wenn ein Ereignis wahr ist, dann soll eine Handlung1 ausgeführt werden. Ist dies nicht der Fall (ansonsten) soll eine Handlung2 ausgeführt werden“.

Diesen Befehl findest du unter dem Block „Logik“ und dort „wenn wahr dann --- ansonsten“. Docke diesen Befehl in dein Programm ein.

Ich habe für euch schon mal das Gesamtprogramm geschrieben und festgestellt, dass mit einem analogen Wert von 500 der Fotowiderstand schaltet. Dieser Wert muss im Programm abgefragt werden. Den entsprechenden Befehl findest du unter dem Block „Logik“.

Jetzt baust du die Abfrage „wenn „analoger Wert von Pin P1“ „< kleiner“ „500“ ist ins Programm ein.

Hier kannst du den Befehl duplizieren und ihn an der richtigen Stelle einfügen.

Füge für „dann“ den Befehl

„Schreibe digitalen Wert von .... P15“ ein. Ändere auf „1“ um. Dupliziere den Befehl und füge ihn unter „ansonsten“ ein. Ändere auf „0“ um.

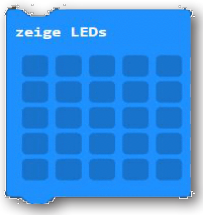
Somit ist die Aufgabe 1 gelöst. Speichere und teste das Programm.

Händetrockner mit LED-Anzeige

Hier noch eine Zusatzaufgabe die du lösen musst.

ventilator2.hex

Auf deinem micro:bit befindet sich eine LED-Matrix mit 25 LED´s. Mit dieser sollst du den jeweiligen Schaltzustand des Motors anzeigen. Wenn der Motor läuft, soll ein + erscheinen ansonsten ist die Matrix aus.

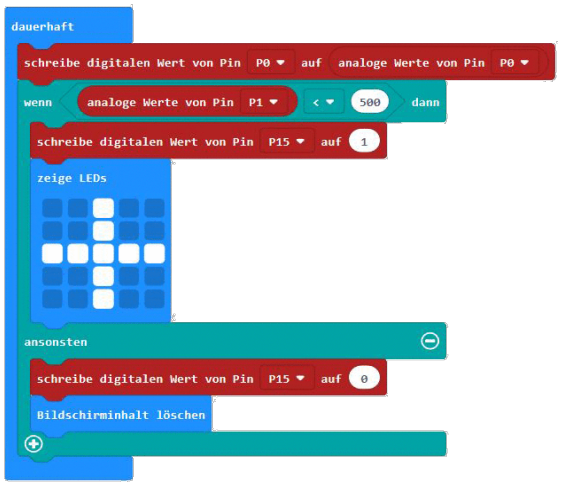


Das geht eigentlich recht einfach. Im Block „Grundlagen“ findest du den Befehl „zeige LEDs“. Ziehe diesen vor den Befehl „ansonsten“.

Klicke mit der Maus die entsprechenden LED´s an. Diese werden weiß dargestellt.

Füge in „ansonsten“ den Befehl „Bildschirminhalt löschen“ aus dem Block „Grundlagen“ und „... more“ ein.

Auch das wäre geschafft. Speichere das Programm nochmal ab und teste es an deinem Modell.



Schranke

In vielen Städten findest du Parkhäuser oder große Parkplätze, die mit einer Parkhausschranke das Ein- und Ausfahren der Fahrzeuge steuern. Ist z.B. ein Parkhaus komplett belegt, wird dies durch ein Parkleitsystem angezeigt. An verschiedenen Straßen findest du elektronische Anzeigen, die dir sagen, in welchem Parkhaus noch Kapazitäten frei sind oder welche belegt sind.

Es gibt verschiedene Möglichkeiten eine Schranke zu aktivieren.

Z.B. über die Eingabe eines Zahlencodes, einer Codekarte aber auch durch eine Lichtschranke, wie du sie später in deinem Modell vorfindest.

Stelle dir auch für diese Aufgabe folgende Fragen:

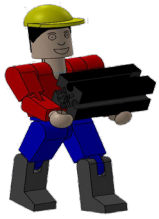
- Wann öffnet die Schranke?
- Leuchtet eventuelle eine Kontrollanzeige?
- Wann schließt die Schranke wieder?
- Wie lange ist die Schranke offen?

Notiere dir einfach die Lösungen auf einem Blatt Papier und verwende die Ergebnisse später bei den einzelnen Programmbefehlen.

Auch hier viel Spaß beim Lösen dieser Aufgabe.

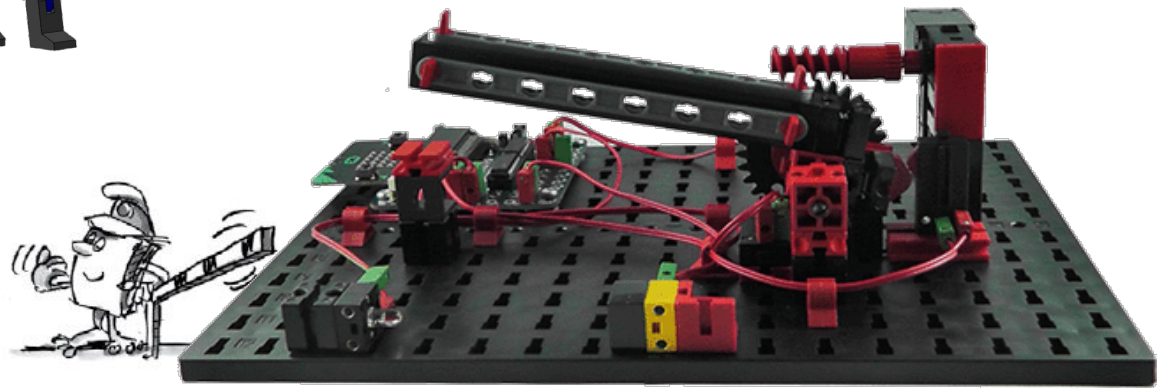
Dein RoBo

Parkhausschranke



Baue anhand der Bauanleitung dein drittes Modell auf.

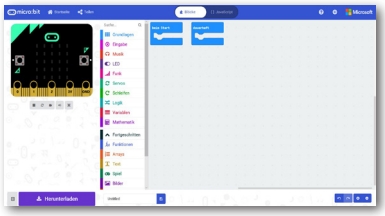
Verdrahte das Modell entsprechend des Schaltplanes.



Auch für dieses Modell werde ich dir eine Aufgabe, und da du ja schon ein Programmierprofi bist, drei Zusatzaufgaben stellen, die du dann in ein Steuerprogramm umsetzen kannst.

schranke1.hex

Fährt ein Fahrzeug in die Lichtschranke und unterbricht den Lichtstrom, soll die Schranke nach 1 Sekunde öffnen. Sie soll 5 Sekunden geöffnet bleiben und dann wieder schließen.



Auch bei dieser Aufgabe werde ich dir natürlich helfen.

Melde dich im Internet an und starte wie bei den vorherigen Aufgaben deinen Editor.

Wichtig: Stelle auch hier wieder sicher, dass du den „Fortgeschrittenen“ Block mit der entsprechenden Schaltfläche umgestellt hast.

Du fängst wieder mit dem Programmteil „beim Start“ an.

Damit das Programm später richtig funktioniert, musst du zuerst den Befehl „LED aktivieren aus dem Block „Led ...more“ einfügen

Zu Beginn des Programms sollen die Motoranschlüsse am „P16“ und „P15“ auf „0“ gesetzt werden.

Am „Pin13“ hast du die LED deiner Lichtschranke angeschlossen. Diese soll nach Programmstart leuchten. Füge die Befehle ein und ändere ihre Eigenschaften.

Der Hauptteil des Programms wird wieder mit dem Befehl „dauerhaft“ begonnen.

Schau dir noch einmal das Programm des Händetrockners an. Bei diesem Programm fragst du die Lichtschranke ab und ermittelst den Wert des Fototransistors. Diese Befehlsfolge kannst du auch bei der Schranke nutzen.

Ziehe wie bei dem Händetrockner die entsprechenden Befehl in dein Programm. Der Fototransistor wird auch bei der Schranke am „Pin1“ angeschlossen. Ändere entsprechend den Wert um.

Der nächste Befehl ist die „wenn ..dann“-Abfrage aus dem Block „Logik“. Auf die Raute „wahr“ ziehst du den Befehl einer Abfrage ein. Ändere den Wert nach dem „<“ auf 500 um.

Wenn der Wert der Lichtschranke kleiner 500 ist, soll die Schranke öffnen. Hierfür fügst du für „dann“ 2 mal „schreibe digitalen Wert von ....“ ein. Ändere den ersten Befehl auf „P16“ und auf „1“ und den zweiten Befehl auf „P15“ und auf „0“ um.

Wird das Programm gestartet, öffnet die Schranke bzw. der Motor startet. Dieser soll so lange laufen, bis der Taster am „Pin3“ geschlossen wird.

Dazu benötigst du aus dem Block „Schleifen“ den Befehl „während ... mache“. Ziehe diesen unter den letzten Befehl.

Ersetzt die Raute wahr“ wieder mit einer Abfrage aus dem Block „Logik“.

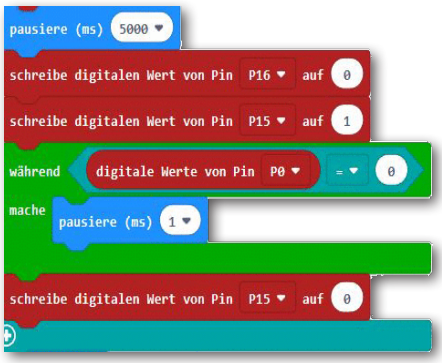
Füge als erste Variable den Befehl „digitale Werte von ...“ ein. Ändere die Pinbezeichnung auf „P3“ um.

In die leere Programmstelle fügst du aus dem Block „Grundlagen“ den Befehl „pausiere (ms)“ mit dem Wert von 100 ein.

Teste schon mal das Teilprogramm. Unterbrichst du die Lichtschranke, öffnet die Schranke, betätigt den Schalter und bleibt stehen.

In der Aufgabe steht, dass wenn die Lichtschranke unterbrochen wird, die Schranke nach 1 Sekunde öffnet. Baue diesen Befehl an der richtigen Stelle ein und ändere die Wartezeit auf 1 Sekunde um.

Die Aufgabe verlangt weiter, dass nach 5 Sekunden die Schranke schließt. Dazu baust du nochmals den Befehl „pausiere (ms)“ ein. Anschließend soll der Motor in die andere Richtung drehen bis der Taster am „Pin0“ betätigt wird.



Dupliziere einfach die benötigten Befehle aus dem ersten Programmteil.

Ändere den Wert von „P16“ auf „0“ und den Wert von „P15“ auf „1“. Andere auch den Wert des Tasters von P3“ auf „P0“um

Damit die Schranke geschossen stehen bleibt, musst du noch den letzten Befehl von „P16“ auf „P15“ umändern.

Teste das Gesamtprogramm. Unterbrichst du die Lichtschranke, öffnet die Schranke, betätigt den Schalter und bleibt stehen. Nach einer bestimmten Wartezeit schließt die Schranke wieder.

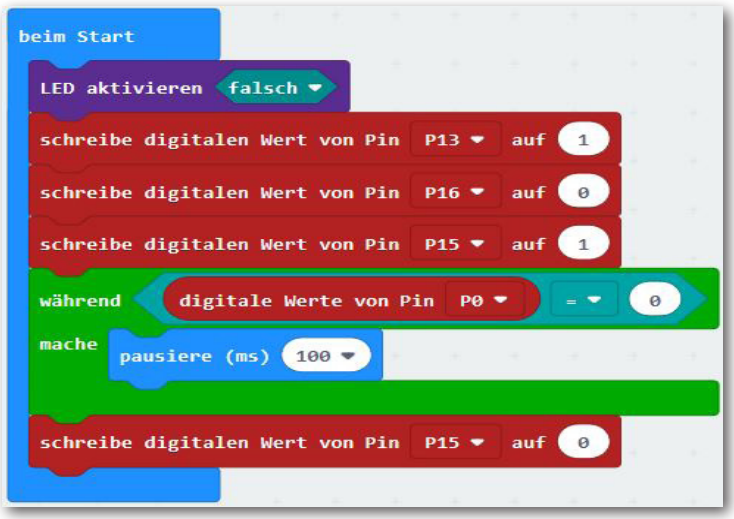
Was geschieht, wenn die Schranke bei Programmstart offen steht? Eigentlich müsste man sie von Hand schließen. Aber warum können wir das nicht durch eine Programmergänzung realisieren?

schranke2.hex

Steht die Schranke beim Programmstart offen, soll diese zuerst geschlossen werden.

Aus dem Programmteil „Dauerhaft“ duplizierst du den Programmblock in das Startprogramm.

Du kannst jetzt das Programm testen. Stelle die Schranke mechanisch in eine Mittelposition. Nachdem das Programm heruntergeladen wurde, schließt zuerst die Schranke und arbeitet dann den Programmteil „dauerhaft“ ab.



Das Gesamtprogramm sieht nun wie folgt aus.

Speichere das Programm auf deinem Rechner ab. Verwende einen neuen Namen, z.B. Schranke2.

Da habe ich noch eine Idee. Was spricht dagegen, wenn du in die Anlage noch einen Serviceschalter (Taster A) einbaust, mit dessen Hilfe die Anlage gestartet wird?

Füge einen Programmteil ein, mit dem du die Anlage startest.

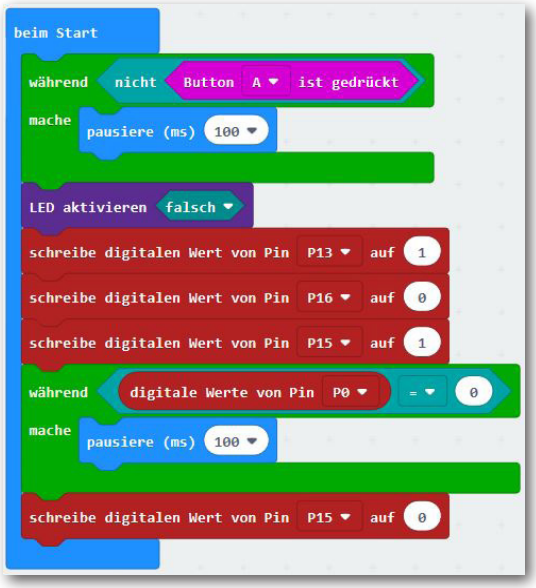
schranke3.hex

Die entsprechenden Befehle fügst du natürlich in den Programmteil „beim Start“ ein. Füge vor der LED-Aktivierung den Befehl „während --- mache“ aus dem Block „Schleife“ ein.

Ersetze die Raute „wahr“ mit dem Befehl „nicht“ aus dem Block „Logik“.

Zur Abfrage des Tasters A verwendest du den Befehl „Botton A ist gedrückt“ aus dem Block „Eingabe“.

Die Lücke für „mache“ ergänzt du mit einem Duplikat von „pausiere (ms) 100“



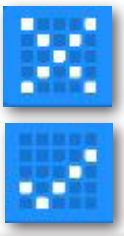
Du kannst jetzt das Programm testen. Stelle die Schranke in eine Mittelposition. Nachdem das Programm heruntergeladen wurde, wartet es auf die Taste A. Wird diese betätigt, schließt zuerst die Schranke und arbeitet dann den Programmteil „dauerhaft“ ab.

Speichere das Programm auf deinem Rechner ab. Verwende einen neuen Namen, z.B. Schranke3.

Parkhausschranke mit optischer Anzeige

Hier habe ich noch eine Programmerweiterung für dich.

schranke4.hex



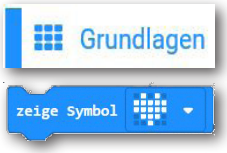
Die Stellung der Schranke soll optisch über das LED-Feld angezeigt werden. Schranke geschlossen ein Kreuz, Schranke offen ein Haken.

Zuerst erweiterst du den „Start-Block“ mit dem Befehl „LED akivieren ....“ aus dem Block LED.

Docke den Befehl über dem „während ... mache“ Befehl ein und ändere den Wert von „falsch“ auf „wahr“ um.

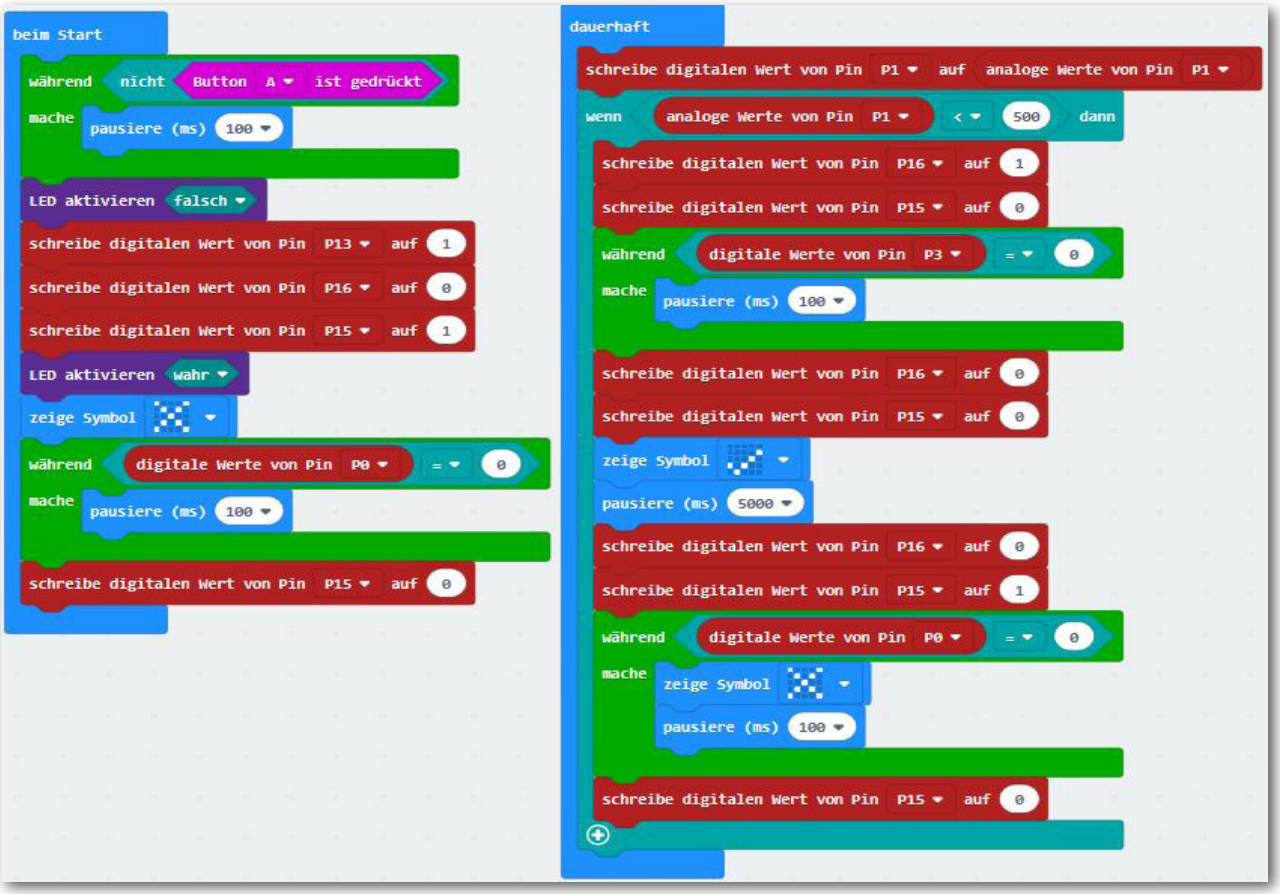
Füge anschließend aus dem Block „Grundlagen“ den Befehl „zeigt Symbol“ ein. Klicke auf den Pfeil und wähle aus der Auswahl die Anzeige eines Kreuzes aus. Diese Befehlsfolge dient dazu, beim Start die Schranke auf geschlossen zu schalten.

Im Programmteil „dauerhaft“ benötigst du den Befehl „zeige Symbole an zwei Stellen. Einmal wenn die Schranke offen ist und dann noch einmal wenn sie geschlossen ist. Füge den Befehl ein und ändere die LED-Anzeige.



Teste das Programm mit der Erweiterung. Stelle die Schranke in eine Mittelposition. Nachdem das Programm heruntergeladen wurde, wartet es auf die Taste A. Wird diese betätigt, erscheint die grafische Darstellung (X) und die Schranke schließt und arbeitet dann den Programmteil „dauerhaft“ ab. Wird die Lichtschranke unterbrochen, geht die Schranke auf und schaltet für 5 Sekunden die LED-Darstellung auf den Haken um. Geht die Schranke wieder zu, wird auch die LED-Anzeige geändert

Speichere das Programm auf deinem Rechner ab. Verwende einen neuen Namen, z.B. Schranke4.



Wenn etwas nicht funktioniert ...

… findest du in dieser Tabelle hoffentlich eine Lösung für dein Problem.

Problem	Mögliche Ursache	Störungsbehebung
1. Software makecode bekommt keine Verbindung zum micro:bit	USB-Kabel ist nicht verbunden	USB-Kabel einstecken
2. Taster funktioniert nicht	Elektrische Stecker an den falschen Anschlüssen des Tasters oder des mikro:bit eingesteckt	Am Taster die Anschlüsse 1 und 3 verwenden. Am mikro:bit Stecker an den beiden Buchsen für I5, I4 oder I1 einstecken
3. Fototransistor funktioniert nicht	Elektrische Stecker falsch eingesteckt	Am Fototransistor: Roten Stecker auf Seite mit rotem Punkt, grünen Stecker auf Seite ohne Markierung einstecken.
	LED der Lichtschranke leuchtet nicht	LED an I1 und P1 anschließen, auf Polung achten
	LED leuchtet schräg am Fototransistor vorbei	LED so verschieben, dass der Fototransistor beleuchtet wird
4. Motor dreht sich nicht	Motor nicht am mikro:bit angeschlossen	Motor wie im Schaltplan des jeweiligen Modells beschrieben am mikro:bit anschließen
	Motor am falschen Motorausgang des mikro:bit angeschlossen	Mit Schaltplan prüfen an welchen Ausgang M1 der Motor gehört und mit diesem Ausgang verbinden
5. Motor dreht sich in falsche Richtung	Bei elektrischen Steckern rot und grün vertauscht	Roten und grünen Stecker am Motor vertauschen
		Im Steuerprogramm die Drehrichtung des Motors ändern Pin15/Pin16
8. Problem hier nicht beschrieben	Nicht gefunden	Wende dich direkt an fischertechnik, z. B. über: **www.fischertechnik.de**

Zum Schluss eine wichtige Internetadresse. Möchtest du mehr über den micro:bit erfahren findest du dies unter

https://microbit.org

So, das war es. Ich wünsche dir viel Erfolg bei der Programmierung der ft-Modelle mit makecode.

1. Auflage 2019

fischertechnik GmbH
Klaus-Fischer-Straße 1
72178 Waldachtal

Tel: (+49) 7443 12 - 4369
E-Mail: info@fischertechnik.de

Hermann Weininger,
Fachoberlehrer und
Elektromeister