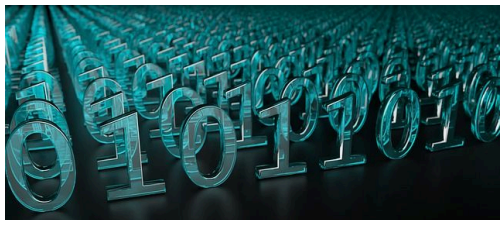
Von der Idee zum Programm



Die meisten Programme werden nicht einfach drauflos geschrieben, sondern sind das Ergebnis eines langen, sowohl kreativen als auch sorgfältig dokumentierten Entwicklungsprozesses. Am Beispiel der Binärzahlen werden wir heute dieses Prinzip nachstellen.

Binär- und Dezimalzahlen

Binärzahlen verwenden nur die Symbole 0 und 1, um Ganzzahlen zu repräsentieren. Dabei wird als Index angegeben, ob es sich um eine Binärzahl (1_2) oder Dezimalzahl (1_{10}) handelt.

Für die Umrechung in eine Dezimalzahl wird die letzte Stelle mit 1, die vorletzte mit 2, die vorvorletzte Stelle mit 4 malgenommen und dann addiert (kurz: n-te Stelle von hinten mit 2^{n-1} malnehmen)

Alternativ kannst Du auch eine Tabelle verwenden (siehe rechts).

Zuordnung bin/dez

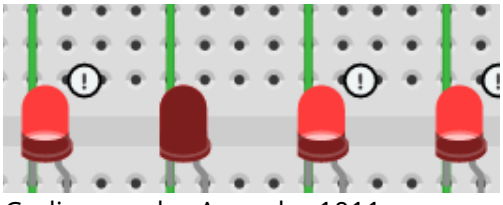
$0_{10} = 000_2$
$1_{10} = 001_2$
$2_{10} = 010_2$
$3_{10} = 011_2$
$4_{10} = 100_2$
$5_{10} = 101_2$
$6_{10} = 110_2$
$7_{10} = 111_2$

Beispiele:
$11_2 = 1 * 2 + 1 * 1 = 3_{10}$
$110_2 = 0 * 1 + 1 * 2 + 1 * 4 = 6_{10}$

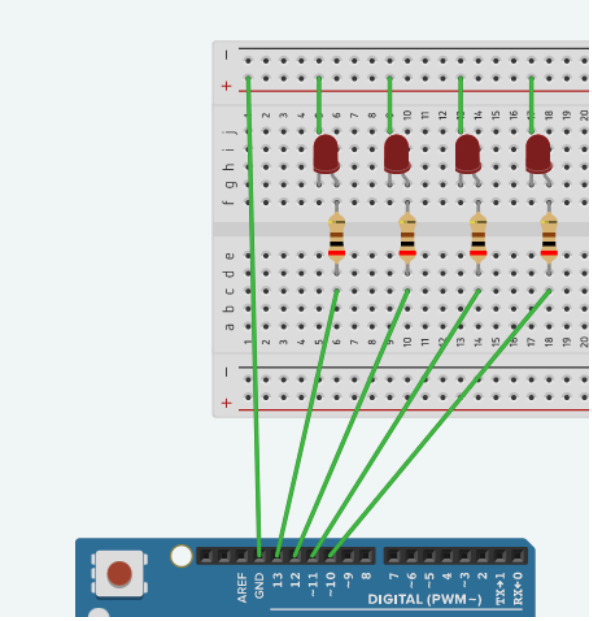
Vorüberlegungen / Entwurf

① Erste Überlegung: Wie können Werte zur Umrechnung ein- und nach der Umrechung ausgegeben werden.

- Die Eingabe werden wir vereinfacht über die Belegung einer Variablen vor Programmstart vornehmen (wenn Du schnell mit der Hauptaufgabe fertig bist, wäre das Erstellen einer Tasterbasierten Eingabe eine lohnende Aufgabe um zu zeigen, was Du kannst).
- Die Ausgabe soll über LEDs gelöst werden, die der Reihe nach abgelesen werden (siehe unten).
- Entwirf einen möglichen Arduino-Aufbau, der das zielgerichtete Ein- und Ausschalten von 4 LEDs ermöglicht.



Codierung der Ausgabe 1011



Vergleiche Deinen Entwurf mit diesem Vorschlag. Prüfe, ob Dein Entwurf die folgenden Kriterien erfüllt:
* 4 LEDs verbaut
* 4 Widerstände an den LEDs verbaut
* Jede LED ist an einem separaten PIN angeschlossen, damit sie separat ansteuerbar ist
* Alle Stromkreise treffen sich auf dem Breadboard und werden gemeinsam zum GND-PIN geschlossen
* Alle Stromkreise sind geschlossen

Hinweis: Die Schaltung muss nicht genauso aussehen wie auf dem Bild, solange sie die genannten Kriterien erfüllt.

Algorithmus

Bevor Du versuchst, ein Programm zu schreiben, solltest Du einen Plan für den Ablauf machen und diesen testen. Es gibt verschiedene Wege, um Dezimalzahlen in Binärzahlen umzuwandeln; zwei davon stelle ich Dir kurz vor:

Umwandlung durch Einzelfall-Identifika...
Prüfe, ob die umzuwandelnde Zahl 0 ist, 1 ist, 2 ist, ... und hinterlege für jede denkbare Dezimalzahl bis 7 die passenden HIGH/LOW-Werte für die LED-Pins.

Achtung: Das ist zwar einfach, aber nicht gut zu erweitern.

Umwandlung durch Division mit Rest
Teile die Eingabe durch 2; der Divisionsrest ist die letzte Stelle der Binärzahl. Das ganzzahlige Ergebnis der Division ist die neue Zahl. Diese wird durch 2 geteilt, der Divisionsrest gibt die vorletzte Stelle der Binärzahl an (usw., bis der ganzzahlige Rest 0 ist).

Beispiel:
5 : 2 = 2 Rest 1
2 : 2 = 1 Rest 0
1 : 2 = 0 Rest 1
-> Ergebnis: 101_2

② Erstelle zunächst als Text einen Algorithmus zur Umwandlung einer Dezimalzahl zwischen 0 und 7 in eine Binärzahl. Das Ergebnis soll über die geplante Arduino-Schaltung ausgegeben werden.

- Beginne mit „SPEICHERE Zahl ALS Eingabe“.
- Verwende ausschließlich die folgenden Textbausteine: WENN (Bedingung) DANN Anweisung (ggf: ANSONSTEN Anweisung), MERKE Wert ALS Eingabe, ist gleich / ist ungleich, Zahlen, Rechenoperationen \+ - * DIV MOD, PIN Nummer, STROM EIN/AUS AN Pin..., WIEDERHOLE BIS / n-mal
- Hinweis: MOD liefert den Divisionsrest (5 MOD 2 = 1)
- Hinweis: DIV ersetzt das Geteiltzeichen, gibt aber nur den Ganzzahligen Anteil des Ergebnisses an (5 DIV 2 = 2)
- Achtung: Nicht alle diese Bausteine werden notwendigerweise benötigt!

Testen / Simulieren des Algorithmus

Jetzt soll getestet werden. Dabei übernehmen Mitschüler*innen die Rollen der beteiligten Komponenten (der LEDs, der Recheneinheit, der Kontrolleinheit, der Variablen).

Rollenbeschreibungen

Kontrolleinheit

Du liest Schritt für Schritt die Anweisungen vor - NICHT INTERPRETIEREN, sonst werden Fehler nicht gefunden. Wenn Du eine Rechnung durchführen oder eine Bedingung überprüfen lassen möchtest, beauftragst Du die Recheneinheit, das Merken/Wiedergeben von Zahlenwerten übernimmt die Variable.

Recheneinheit

Du wirst von der Kontrolleinheit beauftragt, entweder Rechnungen durchzuführen (dann meldest Du das Ergebnis zurück) oder Bedingungen zu prüfen (dann melde WAHR oder FALSCH zurück).
Beachte: Es ist wichtig, dass Du genau das Gefragte und NUR das Gefragte lieferst.

LED (viermal)

Du bildest mit den anderen LEDs eine Ausgabereihe. Es ist wichtig, dass Du Deine PIN-Nummer kennst. Wenn an Deinem PIN der Strom eingeschaltet wird, stehst Du auf und bleibst stehen, bis der Strom an Deinem PIN wieder ausgeschaltet wird. Du startest im Zustand „kein Strom“.

Variable „Eingabe“

Du merkst Dir Zahlenwerte, die Dir die Kontrolleinheit übergibt. Wenn Kontrolleinheit oder Recheneinheit einen Variablenwert abfragen, gibst Du ihn an.
Beachte: Auch wenn die Variable „Eingabe“ heißt, kann sich der Wert im Verlauf der Ausführung des Algorithmus ändern.

Testlauf

③ Spielt den Programmablauf für verschiedene Startwerte der Variablen Eingabe durch und prüft, ob das Ergebnis korrekt ist. Ggf. muss die Anleitung angepasst werden. Wechselt euch fair ab, wenn es um die getesteten Programme geht, damit jeder sein Ergebnis prüfen und ggf. verbessern kann.

④ Startet ggf. weitere Testläufe, bis die meisten Algorithmen funktionieren.

⑤ Falls Du früh fertig bist: Überlege, wie Dein Programm für 4 statt 3 Ausgabe-LEDs geändert werden müsste und wie sich das auf mögliche Eingaben auswirkt.

Endlich: Umsetzung

⑥ Setze Deinen Algorithmus als Arduino-Programm um. Wenn Du sehr schnell bist, kannst Du als Erweiterung eine tasterbasierte Eingabe planen und umsetzen.