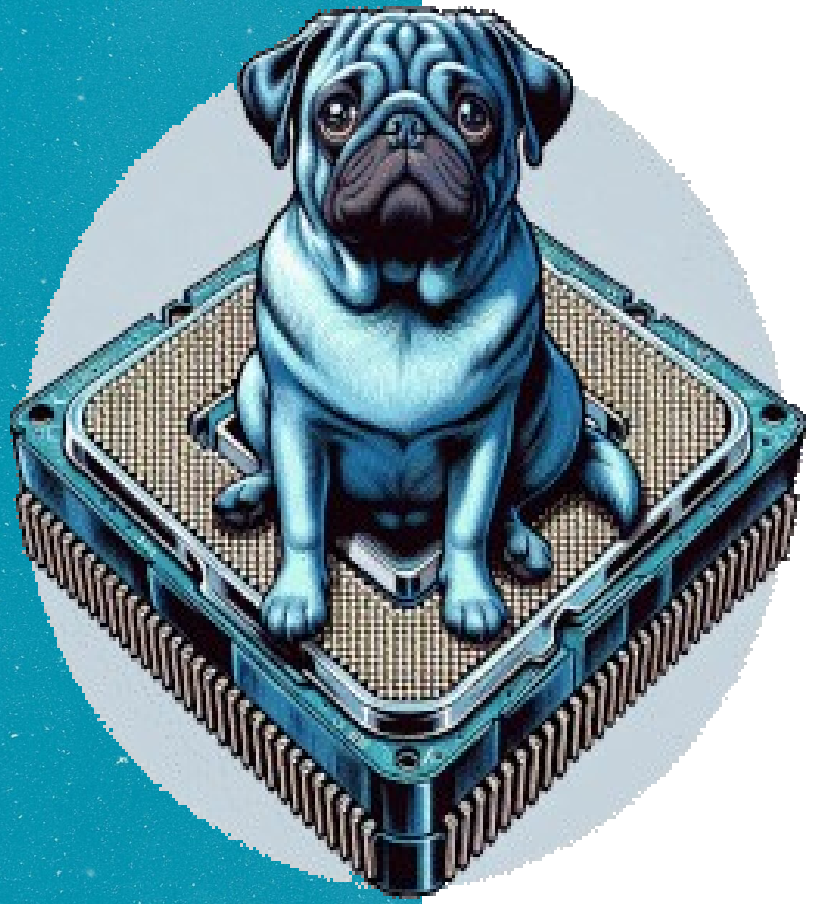
Mikrocomputertechnik

Ein Fach für angehende Programmierer an der höheren Fachschule TEKO

WS25, Stefan Mühlebach

Das Ziel

Die Studierenden können ein
Mikroprozessor- resp. ein Mikrocontrollersystem
mit geeigneter Programmiersprache programmieren.



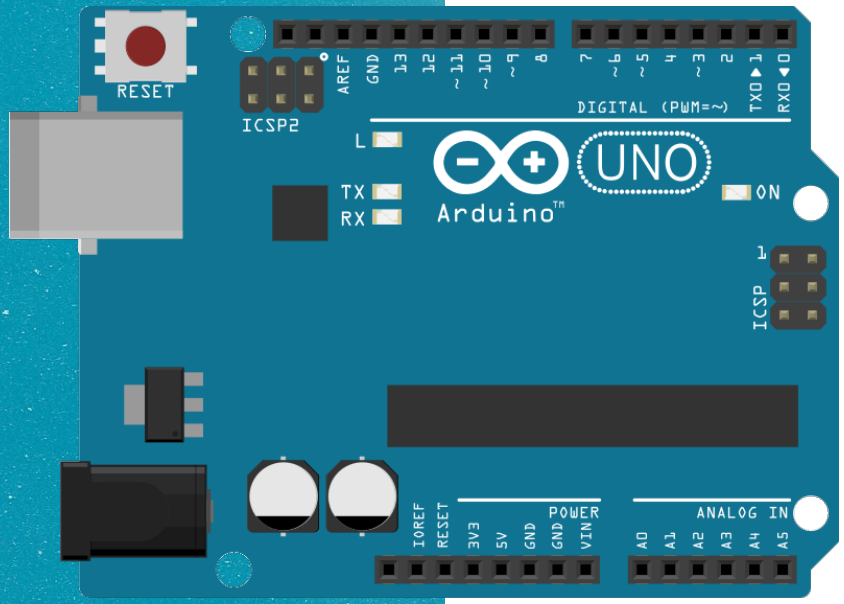
MOPS (Modellprozessor)

Virtuell, sehr einfach

Die richtige Umgebung, um erste (und wohl auch letzte) Schritte mit *Assembler* zu machen.

Algorithmen, mal anders

Ihr werdet mit einige Algorithmen programmieren und euch dabei an *sehr wenig Platz* und noch *viel weniger Komfort* gewöhnen müssen.



Arduino

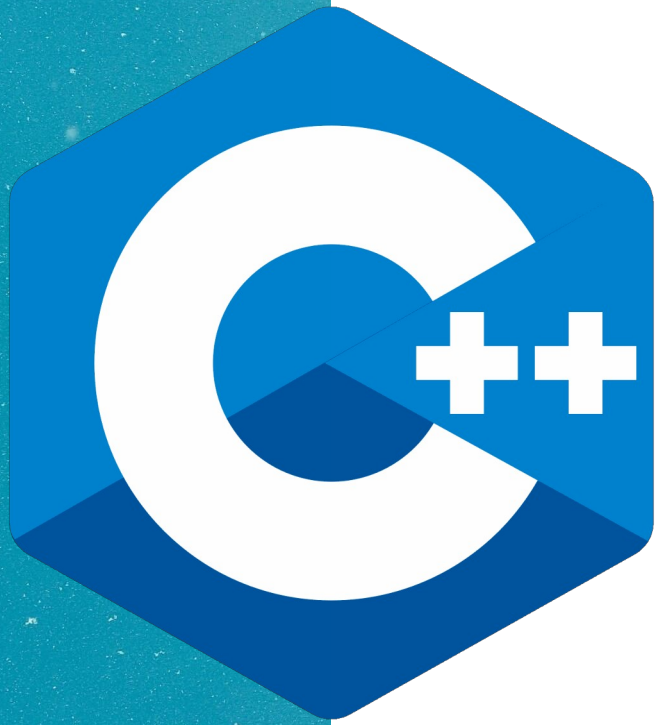
Ein ganzes Ökosystem

Nicht nur gehören mittlerweile mehrere Boards dazu, sondern auch Libraries, eine unüberblickbare Menge an Sensoren, Aktoren und Peripheriegeräten.

URI (oder U=R*I)

Mit dieser Welt verbunden ist auch die Elektro-, resp. Digitaltechnik und ich werde euch die wichtigsten Formeln, Handhabungen und Hintergründe dazu vermitteln.

C++



Eine neue Prog.sprache

In der Arduino-Welt ist C++ die erste und keine schlechte Wahl. Es gibt grosse Parallelen zu C#, aber auch sehr viele Unterschiede.

Klassen und Objekte

Mein Ziel ist auch, mit euch eigene Klassen zu erstellen und dabei sich weiter eingehend mit guter OO-Programmierung zu befassen.

Organisatorisches

Gewichtung

Mit MOPS: 2 Nachmittag
Der Rest gehört Arduino

Lehrmittel

Einstieg in die Arduino-Welt
mit einem Lern-Comic

Prüfungen

Eine neue Idee dazu

OnlOffl

Aufgrund der Abhängigkeit
zu Hardware finden alle 9
Nachmittage vor Ort statt

Wie kommen wir zur Fachnote?

Arbeiten

Alle Arbeiten von euch werdet ihr auf GitHub ablegen (privat) und mir Zugriff gewähren. Die schaue ich mir an… die erste Note

Erfahrungsnote

Ich werde mir eure Beiträge, euer Miteinander, der Umgang mit der Hardware, etc. verfolgen und am letzten Tag in Absprache mit euch die zweite Note bauen.

Es geht los!

Ihr findet in der Dateiablage in Teams einen Ordner mit «MOPS» im Namen und darin mehrere Dateien. Ihr braucht einerseits die ZIP-Datei, welche MOPS beinhaltet und andererseits die zu eurer HW/OS passende Python-Variante.

Wichtig: ihr müsst Version 3.11 installieren – egal was ihr davon haltet.

Algorithmus 0: Addition zweier Zahlen

Lasse den Benutzer zwei Zahlen über das Eingaberegister eingeben, addiere sie und gib das Resultat über das Ausgaberegister wieder aus.

Algorithmus 1: grösster gemeinsamer Teiler

Der grösste gemeinsame Teiler (ggT oder engl. gcd -- greatest common divider) zweier natürlichen Zahlen *a* und *b* wird nach folgendem Algorithmus (dem Euklidischen Algorithmus) berechnet:

1) Wenn *a* gleich *b* ist, sind wir fertig, das Resultat ist gleich *a*.
2) Wenn *a*>*b* ist, ersetze *a* durch *a*-*b*, sonst *b* durch *b*-*a*.
3) Gehe zu Schritt 1.

Erstelle ein Programm, welches den ggT von zwei Zahlen berechnet und ausgibt.

Beispiel: ggT(1071, 462) = 21

Algorithmus 2: Fakultät

Die Fakultät einer natürlichen Zahl *n* wird berechnet durch:

$n! = 1 \cdot 2 \cdot 3 \cdot 4 \ldots (n-1) \cdot n$

Erstelle ein Programm, welches zu einer beliebigen Zahl die Fakultät berechnet.

Beispiel: 5! = 120

Algorithmus 3: Zufallszahlen

Zufallszahlen sind in keinem Computer wirklich zufällig, sondern werden meistens berechnet – sind also vorhersehbar. Die bekannteste Methode ist die der Linearen Kongruenz. Ausgehend von einem Startwert X_0 werden die weiteren Werte wie folgt berechnet:

$$X_{n+1} = (aX_n + c) \bmod m$$

Wobei die Werte für a, c und m fix sind. Reale Werte sind meist ausserordentlich gross und für uns nicht geeignet. Versucht es daher mit $a=157$, $c=3$ und $m=256$ und als Startwert $X_0=233$. Der Algorithmus wartet auf eine Eingabe vom Benutzer (die allerdings nicht verwertet wird) und liefert als Output die nächste Zufallszahl.

Algorithmus 4: Primzahlen-Check

Im letzten Algorithmus soll MOPS die Eingabe des Benutzers dahingehend prüfen, ob es eine Primzahl ist. Primzahlen sind alle ganzen Zahlen grösser als 1, welche nur durch sich selber und durch 1 ganz teilbar sind.

Falls es eine Primzahl ist, soll der Algorithmus eine 1 ausgeben, andernfalls eine 0.

Primzahlen sind beispielsweise: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29 aber auch 601, 607, 613, 617, etc.

Links

Eine weitere, einfache und interaktive CPU-Simulation
https://cpuvisualsimulator.github.io/

Ein weitaus potenterer CPU- und OS-Simulator
https://teach-sim.com/

Oder der MIPS-Visualizer (MIPS ist eine reale CPU)
https://aleksa-sukovic.github.io/mips-visualiser/

Der Zufallszahlen-Algorithmus
https://en.wikipedia.org/wiki/Linear_congruential_generator

Merci!

Stefan Mühlebach, TEKO