

**FRIEDRICH-SCHILLER-
UNIVERSITÄT JENA**



seit 1558

JENAER SCHRIFTEN
ZUR
MATHEMATIK UND INFORMATIK

Eingang: 22.02.2021 Math/Inf/01/2021 Als Manuskript gedruckt

**Durchführung von Projekten der technischen Informatik
mit Design-based Learning**

Andreas Kiener
Private Pädagogische Hochschule der Diözese Linz
Salesianumweg 3, A-4020 Linz

Durchführung von Projekten der technischen Informatik mit Design-based Learning

Andreas Kiener

Private Pädagogische Hochschule der Diözese Linz

Salesianumweg 3

A-4020 Linz

Inhaltsverzeichnis

1 Einleitung	3
2 Didaktischer Hintergrund	4
3 Projektmanagement mit SCRUM.....	8
3 Durchführung	11
4 Material	12
5 Projekte	16
5.1 Hot Wire.....	17
5.2 Mars Rover.....	20
5.3 Anemometer.....	23
5.4 Anemoskop – Windrichtungsgeber	26
5.5 Schrittzähler.....	29
Literaturverzeichnis	32
Abbildungsverzeichnis.....	35
Tabellenverzeichnis	36
Anhang	37

1 Einleitung

Ubiquitous und Pervasive Computing bezeichnen die Entwicklung der immer kleiner werdenden Mikrokontroller, vielfach vernetzt und ausgestattet mit Sensoren. Unbeachtet sind sie mittlerweile Bestandteil vieler Produkte und geben ausreichend Anhaltspunkte für einen Unterricht der technischen Informatik. Für Schülerinnen und Schüler griffigere und auch gebräuchlichere Begriffe sind Internet of Things, Wearable Computing aber auch Physical Computing.

Dieser Entwicklung von Informatiksystemen wird von den drei Grunderfahrungen des Informatikunterrichts Rechnung getragen, indem sie die Erfahrungen, welche Schülerinnen und Schüler in der Auseinandersetzung mit Informatiksystemen machen, widerspiegeln. Ein Informatikunterricht ist dann allgemeinbildend, wenn er ermöglicht, die Wirkungen von Informatiksystemen in unterschiedlichen Lebensbereichen zu entdecken und zu erkennen, dass Informatiksysteme aufgrund von Menschen gestalteter Algorithmen funktionieren und erlernte Problemlösefähigkeiten angewendet werden können [1, 2].

Seit der Markteinführung des Micro:bit sind Mikrokontroller endgültig im Unterricht angekommen, sogar in der Primarstufe [3]. Jedoch gab es schon lange zuvor viele Anregungen zu einem Unterricht mit dem Arduino, auch deshalb, weil er der erste zur Verfügung stehende Mikrokontroller war, der sich für einen Informatikunterricht eignete. Allen vorangegangenen Unterrichtsbeispielen ist gemeinsam, dass jeweils nur ein Mikrokontroller-System eingesetzt wird obwohl es mittlerweile mehrere und auch speziell für die Schule angepasste Mikrokontroller gibt [4, 5].

Ziel dieser Arbeit ist es, Projekte zu Physical Computing sowie Wearable Computing aber auch zur Robotik zu entwerfen, um Schülerinnen und Schülern die Grunderfahrungen des Informatikunterrichts zu ermöglichen.

2 Didaktischer Hintergrund

Nach Andersons und Krathwohls [6] Revision der Bloomschen Taxonomie der kognitiven Lernziele, in der „Creation“ als höchstes anzustrebendes Lernziel gesetzt wurde, kommt der Gestaltung von Informatiksystemen eine besondere Bedeutung zu. Als Unterpunkte zu Creation werden noch Generating, Planning und Producing angeführt. In der gezeigten Darstellung erlaubt das Erreichen des höchstwertigsten Lernziels auch den größten Überblick, insbesondere den Blick auf andere Fächer, da das Kreieren und Gestalten interdisziplinäre Akzente beinhaltet.

Im Fächerkanon der Sekundarstufe nimmt Informatik eine besondere Rolle ein, da es im Gegensatz zu den etablierten NAWI-Fächern zusätzlich noch eine ingenieurwissenschaftliche bzw. Engineering Charakteristik aufweist. Aufgrund dieser inhärenten Eigenschaft des Faches wird es Schülerinnen und Schülern in der Allgemeinbildung im Informatikunterricht und im Besonderen im Unterricht der technischen Informatik ermöglicht, Erfahrungen in der Gestaltung von technischen Geräten zu sammeln, welche ansonsten nur der Berufsausbildung von Ingenieuren vorbehalten ist. An die Stelle des Entdeckens wie in naturwissenschaftlichen Fächern tritt hier das Erfinden.

Auf die Mehrschichtigkeit des Faches Informatik geht Denning [7] ein und führt Mechanics und Design als die „Great Principles of Computing“ an, woran die unterschiedlichen Denkweisen anschließen und daraus abgeleitet die sich für einen Informatikunterricht ergebenden Unterrichtsmethoden.

Tabelle 1 Gegenüberstellung der Great Principles of Computing von Denning zu Denkweisen und Methoden eines Informatikunterrichts.

	Prinzipien	Denkweise	Unterrichtsmethode
GREAT PRINCIPLES OF COMPUTING	Mechanics	Computational	Problem-based
	<i>computation, communication, coordination, recollection</i>	Thinking	Learning
	Design	Design Thinking	Design-based
	<i>simplicity, performance, reliability, evolvability</i>		Learning

Studien der Vergangenheit haben gezeigt, dass es eine Tendenz zu einem domänenspezifischen Denken gibt [8]. In Tabelle 1 sind die Great Principles of Computing den

Denkweisen in der Informatik und den daraus folgenden Unterrichtsmethoden gegenübergestellt.

Jannette Wing prägte den Begriff Computational Thinking als eine universell anwendbare Denkweise der Computerwissenschaften, welche nicht nur von Wissenschaftlern erlernt und eingesetzt werden sollte [9]. Effizient Probleme zu lösen sowie das Entwerfen von Informatiksystemen umreißen dabei das Prinzip von Computational Thinking. Diese Sichtweise, dass Computational Thinking das Formulieren von Problemen ist, um die Lösung als Rechenschritte bzw. Algorithmen zu repräsentieren, wird von Aho [10] unterstrichen.

Somit reflektiert einerseits Computational Thinking die von Denning in den Great Principles of Computing unter Mechanics, wie Computation, Communication, Coordination und Recollection, angeführten Eigenschaften [11]. Andererseits basiert Computational Thinking auf dem Formulieren und Lösen von Problemen, womit sich Problem-based Learning anbietet, um Computational Thinking im Informatikunterricht zu unterrichten [12].

Bedeutender für einen Unterricht zur technischen Informatik ist aber das Prinzip des Designs. Design wird hier als das Gestalten und Kreieren von Informatiksystemen gesehen und fällt mit dem höchsten anzustrebenden Lernziel „Create“ nach Anderson und Krathwohl zusammen [6]. Auch für dieses Prinzip ist eine domänenspezifische Denkweise erforderlich: das Design Thinking. Es beschreibt den Ablauf, welcher notwendig ist, um zielgerichtet Innovationen zu entwickeln. Design Thinking lässt sich grob in den Problemraum mit den Schritten Beobachten und Synthese sowie den Lösungsraum mit den Schritten Ideen und Prototyping unterteilen [13]. Siehe Tabelle 2. Der Schritt des Prototypings unterscheidet hauptsächlich Design Thinking vom Computational Thinking, sie zeigen aber ansonsten große Ähnlichkeiten [14]. Design Thinking wird auch als Synthese von analytischem Denken bei hoher Reliabilität und einem intuitiven Denken bei hoher Validität gesehen [15].

Tabelle 2 Der iterative Prozess von Design Thinking [13]

Problemraum		Lösungsraum	
1. Beobachten	2. Synthese	3. Ideen	4. Prototyping

Design-based Learning ist mit den Charakteristika Design Elements, Projekt Charakteristik, Lehrerrolle, Assessment und sozialer Kontext das Pendant zu Design Thinking. Zu Design Elements werden die Eigenschaften Explore problem und iterative Design Methoden

genannt, welche beide eine Koinzidenz bezüglich dem Problemraum als auch dem Lösungsraum mit Design Thinking aufweisen [16].

Tabelle 3 Dimensionen und Charakteristika von Design-based Learning

Design Elements	Project Characteristics	Social Context	Teacher Role	Assesment
Explore problem, Iterative design methodology	Authentic, Hands on, Multidisciplinary, Open-ended	Collaborative, Peer to peer, Competitions	Coaching on task, Process and self development	Formative, Summative

Weitere Tätigkeiten, welche für Design Thinking und für Design-based Learning wesentlich sind [17]:

- Die Problemdarstellung untersuchen, erkunden
- Nach Alternativen suchen
- Eine funktionale Zerlegung durchführen, vereinfachen
- Nach einer grafischen Darstellung suchen
- Limitierungen und Einschränkungen feststellen, z.B.: Grenzen des Materials
- Vorhandene Designs untersuchen
- Sich in die Benutzerperspektive hineindenken
- Ein maßstäbliches Modell bzw. einen Prototyp bauen
- Der Frage nach Überprüfbarkeit nachgehen
- Eine Fehleranalyse durchführen
- Über den Designprozess reflektieren

In der oben angeführten Liste sind einerseits Tätigkeiten genannt, welche nicht nur für Design Thinking gelten, wie etwa die Problemstellung zu erkunden oder auch diese in überschaubarere Einheiten zu zerlegen sowie über den Prozess zu reflektieren. Sehr spezifisch für Design Thinking ist aber die Grenzen, Belastbarkeit und Limitierungen des Materials festzustellen. In einem Unterricht zur technischen Informatik sind es genau die Eigenschaften der unterschiedlichen Mikrokontroller, welche die Gestaltungsmöglichkeiten eingrenzen. Deshalb ist es gerade für Design-based Learning entscheidend, den Lernenden mehrere Systeme für die Entwicklung eines Prototyps anzubieten und sie selbst entscheiden zu lassen, welcher Mikrokontroller sich für das aktuelle Projekt gerade am besten eignet.

Es gibt mittlerweile viele Unterrichtsbeispiele für Anwendungen mit Mikrocontrollern. Sowohl mit dem Arduino - dem ältesten Produkt, sowie dem Micro:bit, welcher in den vergangenen Jahren große Verbreitung gefunden hat und auch mit Anwendungen für den Calliope mini [5, 18]. Ihnen ist gemeinsam, dass immer nur ein einziges Produkt eingesetzt wurde und keine Auswahlmöglichkeit zwischen den Mikrocontrollern für Lernende möglich war. Die präsentierten Unterrichtsvorschläge sind auf einen einzigen Mikrocontroller zugeschnitten. Das hat für die Lehrperson Vorteile, da dadurch der Aufwand für die Vorbereitung reduziert wird. Aber auch bei der Durchführung des Unterrichts selbst, in der Instrukionsphase oder bei der Fehlersuche verringert die Konzentration auf ein System die Belastung der Lehrperson. Zusätzlich muss noch ein doppelter oder sogar dreifacher Bedarf an Geräten für Schülerinnen und Schülern zur Verfügung stehen. Um diese Auswahlmöglichkeit mit begrenzten Mitteln anbieten zu können, muss der Unterricht entsprechend organisiert werden.

Für die Notwendigkeit einer Auswahlmöglichkeit von Mikrocontrollern bei der Realisierung eines Projekts gibt es noch allgemeinere Argumente als die, sich über die Grenzen des Materials zu informieren.

Motivation. Entsprechend der Self Determination Theory nach Deci und Ryan, sind folgende Motive für Menschen zentral: Die Selbstbestimmung, die soziale Eingebundenheit und das Streben nach Kompetenz. Demnach entsteht eine intrinsische Motivation dadurch, wenn Lernenden eine Auswahl angeboten wird und sie daraus frei wählen können [19].

Kompetenz. Auseinandersetzen mit der Funktionsweise und den Möglichkeiten unterschiedlicher Mikrocontroller. Dadurch lernen Schülerinnen und Schüler die spezifischen Eigenschaften der Systeme kennen, müssen sie analysieren und am Ende bewerten, um eine Auswahl treffen zu können. Damit werden auch die in der Taxonomie der Lernziele nach Krathwohl dargestellten Lernziele erreicht [6].

Allgemeinbildung. Für Lernende in der Berufsbildung mag es vielleicht ausreichen, ein einziges Produkt zu erlernen, um meist sehr spezielle Anforderungen erfüllen zu können. Klafki fordert aber für die Allgemeinbildung, Schülerinnen und Schüler von der Fremdbestimmung in die Selbstbestimmung zu führen, ihnen Fähigkeiten zu vermitteln, selbstbestimmt Entscheidungen treffen zu können [20, S. 19]. Gerade im Unterrichtsfach Informatik und der Informations- und Kommunikationstechnologie ist eine ausgeprägte Produktabhängigkeit gegeben, welche von den Herstellern in ihrem

Sinne verteidigt wird. Deshalb ist es notwendig, Schülerinnen und Schülern das Bewusstsein zu vermitteln Alternativen anzudenken und Produktabhängigkeiten nicht als gegeben hinzunehmen.

3 Projektmanagement mit SCRUM

Neben der Dimension des Design Thinking ist auch noch die Projektcharakteristik ein integraler Bestandteil von Design-based Learning.

Ein Projekt ist nach einer Definition von Hänsel ein „Unterricht, in dem Lehrer und Schüler ein echtes Problem in gemeinsamer Anstrengung und in handelnder Auseinandersetzung mit der Wirklichkeit zu lösen suchen, ...“ [21]. Das bedeutet aber nicht, dass mit Design-based Learning ein mehrwöchiger Projektunterricht in vollständiger Ausprägung notwendig ist, sondern projektartiges Lernen in einem Kleinprojekt mit der Dauer von zwei bis sechs Unterrichtseinheiten entspricht diesem ebenfalls [22].

Für die Planung und Durchführung eines projektbasierten Unterrichts bieten sich Projektmanagementmethoden an, in denen die Lehrperson die Rolle des Coaches einnehmen kann. Im klassischen Wasserfallmodell hingegen bleibt die Lehrperson als Projektleiter in der hierarchischen Lehrer-Schüler-Rolle haften. Für Problem-based Learning und im Allgemeinen für Project-based Learning hat sich die Projektmethode Scrum als hilfreich erwiesen [23, 24]. Der Begriff Scrum kommt aus dem Rugby und steht für eine agile und auf Selbstorganisation von Teams ausgerichtete Projektmanagement-Methode, welche ursprünglich aus der Softwareentwicklung kommt und längst nicht mehr darauf beschränkt ist [25].

Um diese Projektmethode durchführen zu können gilt es zunächst einige Begriffe zu klären [26]. Das Scrum-Team besteht aus den Personen bzw. Personengruppen mit folgenden Aufgaben:

Den *Developern* mit breit gefächerten Fähigkeiten. Sie bringen das Projekt inkrementell voran, planen und führen die unmittelbaren Schritte aus und überlegen auch, wie die Qualität zu sichern ist.

Dem *Product Owner*. Er gibt das Produktziel mit dessen Eigenschaften vor und nimmt das fertige Produkt ab.

Dem *Scrum Master*. Er sollte nicht mit einem Projektmanager verwechselt werden. Der Scrum Master ist in der flachen Hierarchie von Scrum den Developern gleichgestellt. Er unterstützt das Team, in dem die Zeitpläne eingehalten werden, moderiert die Besprechungen und sorgt dafür die Kommunikation aufrechtzuerhalten.

Folgende Ereignisse sind für Scrum wesentlich:

Der *Sprint* ist eine Zeitspanne von fester Länge, im Schulbetrieb eine Stunde oder Doppelstunde und schließt unmittelbar an den vorhergegangenen Sprint an. Zu Beginn des Sprints wird in einem *Daily Stand Up* vor einem Board, dem Kanbanboard, vor dem alle Teammitglieder versammelt sind, gemeinsam das Sprintziel festgelegt. Während des Sprints werden die Aufgaben von den Developern ausgeführt. Ein Sprint ist wiederum ein kleines Projekt, deshalb kann im Unterricht ein Projekt, falls es klein ist, auch aus nur einem einzigen Sprint bestehen.

Sprint Review und *Sprint Retrospektive* folgen am Ende eines Sprints. Hier wird das Ergebnis eines Sprints überprüft sowie die Aufgabenliste, Backlog genannt, fallweise revidiert. Zusätzlich sollte in der Retrospektive auch noch der Prozess selbst reflektiert und angepasst werden.

Zusätzlich bestehen noch die Artefakte:

Product Backlog. Es bildet die gesamte Story des Entstehungsprozesses bis zum fertigen Produkt ab und könnte auch als To-Do Liste bezeichnet werden. Das Product Backlog stellt mehr die Sicht des Auftraggebers dar, welche Eigenschaften das Produkt haben sollte.

Sprint Backlog. Es wird vom Team während des Daily Stand Up aus dem Product Backlog für die nächsten Sprints ausgewählt. Es wird erweitert, falls neu auftretende Anforderungen auftreten oder aus dem Product Backlog verfeinert und ist eine auf Teammitglieder zuordenbare To-Do Liste. Allerdings sollte im Sprint pro Teammitglied nur eine bis zwei Aufgaben zugeteilt werden, welche sie selbst ihren Fähigkeiten entsprechend ausgesucht haben. Falls spezifischere oder auch neu erforderliche Tätigkeiten erforderlich sind, kann diese Liste von den Teammitgliedern noch erweitert werden.

Kanbanboard. Dies ist ein Whiteboard mit den vier Spalten für das Backlog (To-Do), dem Sprint Backlog (Next), dem Sprint (Doing), und den erledigten Aufgaben (Done), siehe Abbildung 1.

Für die hier vorgestellten Projekte kann zwischen strukturierten Projekten mit drei Spalten: To-Do (Sprint Backlog), Doing und Done oder angeleiteten Projekten mit den Spalten: To-Do (Product Backlog), Next (Sprint Backlog), Doing und Done ausgewählt werden, siehe Tabelle 4. Eine horizontale Gliederung der Spalten, den Teammitglieder entsprechend, könnte noch vorgenommen werden. Die Teilaufgaben werden auf Post-its notiert und in die entsprechende Spalte geheftet und nach jedem Sprint in der nächstfolgenden Spalte weiter gereiht. Abgearbeitete Post-its sollten in der Spalte Done stehen bleiben, zum einem aus Motivationsgründen, zum anderen wegen der Dokumentation.

Problematisch wird die Verwendung eines Whiteboards bei mehreren gleichzeitig durchgeführten Projekten. In diesem Fall eignen sich webbasierte Kanbanboards wie es bei <http://scrumblr.ca> über einen sehr niederschweligen Zugang angeboten wird. Damit ist es auch möglich das Backlog für mehrere Projekte vorzubereiten. Das virtuelle Kanbanboard kann dann bei einem Daily Stand Up auf einem Smartboard angezeigt und auch interaktiv verändert werden. Farbpunkte erleichtern die Zuordnung der Tätigkeiten zu den Teammitgliedern.

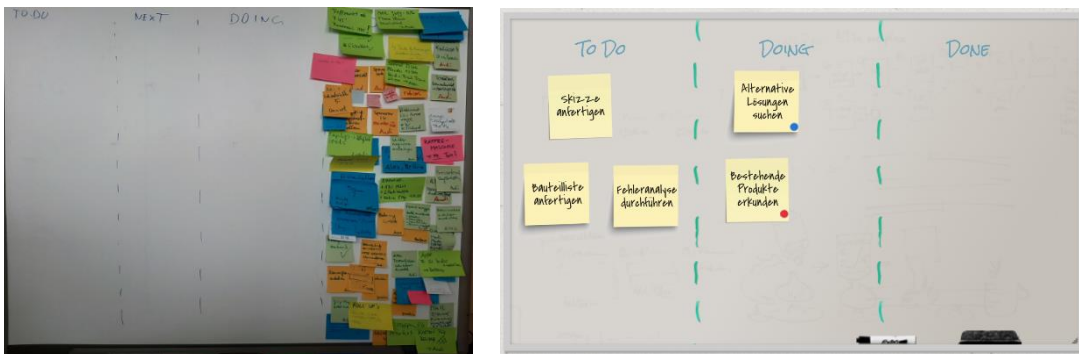


Abbildung 1 - Kanbanboard eines fertig gestellten Projekts und ein Kanbanboard auf scrumblr.ca.
Quelle: Eigene Aufnahmen, CC-BY-SA

3 Durchführung

Für Lehrpersonen ist zunächst die für sie einzunehmende Rolle zu klären. Es gibt zwei Möglichkeiten dafür: Product Owner oder Scrum Master, für beide Rollen gibt es pro und contra Argumente.

Für die Rolle des Product Owners, wie sie auch von eduScrum vorgeschlagen wird spricht, dass die Lehrperson die Aufgaben stellt und letztendlich den fertigen Prototyp abnimmt. Dies entspricht somit genau der Rolle des Product Owners [27]. Der Nachteil aber liegt darin, dass die Lehrperson nicht die Rolle eines Coaches einnehmen kann. Ebenso sind auch die Steuerungsmöglichkeiten für den Lernprozess geringer, obwohl das Product Backlog vorgegeben werden kann.

In der Rolle als Scrum Masters kann die Lehrperson die Termine festlegen, die Kommunikation fördern, ist beim Daily Stand Up, bei den Retrospektiven anwesend, hat Einfluss auf das Sprint Backlog und kann auch beratend den Developern zur Seite stehen. Nachteilig ist der erhöhte Arbeitsaufwand, falls mehrere Teams gleichzeitig an unterschiedlichen Projekten arbeiten.

Aus Untersuchungen zu Inquiry-based und Problem-based Learning weiß man, dass sich ein erhöhtes Maß an Lenkungsmöglichkeiten günstig auf den Ablauf des Projekts auswirkt [28, 29]. Aufgrund deren Ähnlichkeiten zu Design-based Learning kann angenommen werden, dass auch dafür die vier Stufen von Steuerungsmöglichkeiten gelten [30]. Siehe Tabelle 4.

Tabelle 4 Rollenverteilung für Design-based Learning in Abhängigkeit der Strukturierung

	Initiative	Planung	Durchführung	Abschluss
konfirmatorisch	LehrerIn	LehrerIn	LehrerIn	LehrerIn/SuS
strukturiert	LehrerIn	LehrerIn	LehrerIn/SuS	SuS
angeleitet	LehrerIn	LehrerIn/SuS	SuS	SuS
offen	LehrerIn/SuS	SuS	SuS	SuS

Nimmt die Lehrperson die Rolle des Product Owners ein, entspricht es einem angeleiteten Projekt, falls das Product Backlog vorgegeben wird, andernfalls stellt es ein offenes Projekt dar. In der Rolle des Scrum Masters kann das Projekt strukturiert umgesetzt werden, da bei den Daily Stand Ups und den Retrospektiven in der Durchführung eingegriffen werden kann.

Bei der Zusammenstellung der Teams sollte darauf geachtet werden, dass die Teamgröße minimal vier bis maximal sieben Mitglieder beträgt [27, 31]. Bei einer Klassengröße von 28 Schülerinnen und Schülern wäre es daher möglich gleichzeitig vier bis sieben parallellaufende Projekte durchzuführen. Eine zusätzliche Maßnahme, Schülerinnen und Schüler auch paarweise, bekannt als Pair Programming, in einem größeren Team arbeiten zu lassen, könnte die Anzahl der Projekte reduzieren, sie sind dabei zuversichtlicher und auch erfolgreicher [32]. In der Zusammenstellung sollte auch auf differenzierte Fähigkeiten der Teammitglieder geachtet werden.

Für die Abschlusspräsentation haben die Teams idealerweise einen Prototyp vorzuzeigen oder vorzuführen. Bei Interviews nach einem Unterricht von Physical Computing konnte gezeigt werden, dass ein Feedback, welches sie direkt über das fertig gestellte Produkt erhalten, für Schülerinnen und Schüler eine große Bedeutung hat und sich ein Feedback der Lehrperson erübrigt [33].

4 Material

Da in einem Unterricht mit unterschiedlichen und gleichzeitig durchgeführten Projekten eine gemeinsame Instruktionsphase nicht zielführend ist, haben sich Instruktionskarten bzw. Skill Cards in einem Unterricht zu Physical Computing bewährt und waren unkompliziert anzuwenden. Die Gestaltung von Story Sheets erwies sich dabei als schwieriger. [34].

Story Sheets. Für die Projektinitiative wird die gesamte Story eines Produkts benötigt, in strukturierten und angeleiteten Projekten wird sie von der Lehrperson vorgegeben. Die Story wird auf Karten, auch in größeren Formaten möglich, notiert und sollte alle Eigenschaften des Produkts enthalten [24]. Zusätzlich sollte sie genügend Informationen enthalten, um für Schülerinnen und Schüler eine Entscheidungsgrundlage zu bilden, dieses Projekt auszuwählen und um sich näher damit auseinanderzusetzen zu können. Die Story Sheets der vorgestellten Projekte sind im Anhang zu finden und können auch von <https://tutory.de> heruntergeladen werden.

Skill Cards/Lernkarten. Herausfordernd wird die Durchführung mehrerer Projekte mit unterschiedlichen Mikrocontrollern. Eine Instruktionsphase ist wenig sinnvoll, da die Teams unterschiedliche Bedürfnisse an Informationen haben. Deshalb sollten Informationen und Anleitungen auf Blättern, wenn möglich nicht mehr als ein doppelseitig bedrucktes Blatt, mit

allen für die Projektdurchführung relevanten Informationen, zu finden sein. Am Blatt befindliche QR-Codes können zusätzliche Links zu weiteren Informationen enthalten. Im Anhang befinden sich Links zu Skill Cards und Lernkarten.

Post-its/Kärtchen. Auf Post-its oder auch Kärtchen wird das Produkt-Backlog und in detaillierterer Form das Sprint-Backlog geschrieben. Auf je einem Kärtchen sollten die Tätigkeiten möglichst so formuliert sein, dass sie von einem Teammitglied oder einem Paar in einem Sprint ausgeführt werden können. Eine Steuerungsmöglichkeit lässt sich dadurch verwirklichen, dass schon mehr oder weniger spezifische Aufgaben enthalten sind, wie etwa: „OpenRoberta Connector installieren“ oder „Berührung eines Touch-Pins erkennen“.

Baumaterial. Neben den Mikrocontrollern Arduino, Micro:bit und Calliope mini, welche in ausreichender Menge, aber nicht unbedingt in Klassengröße je System vorhanden sein sollen, sind noch Materialien wie Pappe, Drähte, Widerstände, Klebstoffe usw. notwendig.

Software. Als Entwicklungsplattform empfiehlt sich OpenRoberta, damit lassen sich alle drei Mikrocontroller programmieren [35]. Nur eine Plattform zu verwenden ist deshalb sinnvoll, da die Wahl des Mikrocontrollers nicht noch zusätzlich von der Wahl der Software abhängig gemacht werden soll. Für den Micro:bit und den Arduino sind aber in OpenRoberta die Möglichkeiten der Ansteuerung einiger Bauteile beschränkt, deshalb ist die Verwendung deren nativen Entwicklungsumgebungen trotzdem sinnvoll [36].

Werkzeug. Zu den Werkzeugen zählen Computer, die Netzwerkverbindung und Anschlusskabel. Scheren, Klebepistolen, und Akkuschauber sollten ebenfalls vorhanden sein. Für die Fehlersuche ist es von Vorteil, Multimeter zur Verfügung zu stellen.

Microkontroller. Das Herz der Projekte bilden die Microkontroller, siehe Abbildung 2.

Der *Arduino Uno* ist der älteste der drei vorgestellten Microkontroller. Er bezieht seine hervorragende Eigenschaft für einen Informatikunterricht daraus, dass er ursprünglich für Künstler, mit dem Aspekt einer einfach zu bedienenden Hardware entwickelt wurde [37]. Neben 14 Pins, welche sowohl digital als auch pulsweitenmoduliert betrieben werden können, stehen noch sechs analoge Eingänge zur Verfügung. Weitere Sensoren sind on-board nicht vorhanden. Die Besonderheit des Arduinos liegt aber darin, dass es möglich ist sogenannte Shields aufzusetzen, um die Funktionalität beliebig erweitern zu können. Der Arduino kann mittlerweile, ebenfalls so wie alle anderen vorgestellten Produkte, blockbasiert programmiert werden.

Mit dem *Micro:bit* ist seit 2016 der kleinste Mikrokontroller, entwickelt speziell für den Unterricht, auf dem Markt. 21 digitale und analoge Pins sind herausgeführt, aber es ist nur bei 3 Pins möglich direkt mit Klemmen darauf zuzugreifen. Ähnlich wie beim Arduino kann die Funktionalität mit Breakoutboards erweitert werden. Ein Kompass, ein Lagesensor, ein Beschleunigungssensor sowie ein Lichtsensor sind am Mikrokontroller untergebracht, zusätzlich noch zwei Taster und ein 5 x 5 LED-Array. Die Programmierung erfolgt blockbasiert über ein Webinterface, aber auch Phyton ist möglich.

Der jüngste und am besten ausgestattete Mikrokontroller ist der *Calliope mini*, er ist seit 2017 erhältlich. Ganz analog zum Arduino und dem Micro:bit werden Pins herausgeführt, 19 an der Zahl. Der Umfang der Sensoren wird im Vergleich zum Micro:bit um ein Mikrofon erweitert. Ganz entscheidend für Robotikanwendungen sind die am Board befindlichen Motortreiber. Eine Erweiterungsmöglichkeit, wie beim Arduino das Shield und dem Micro:bit das Breakoutboard, ist nicht angedacht.

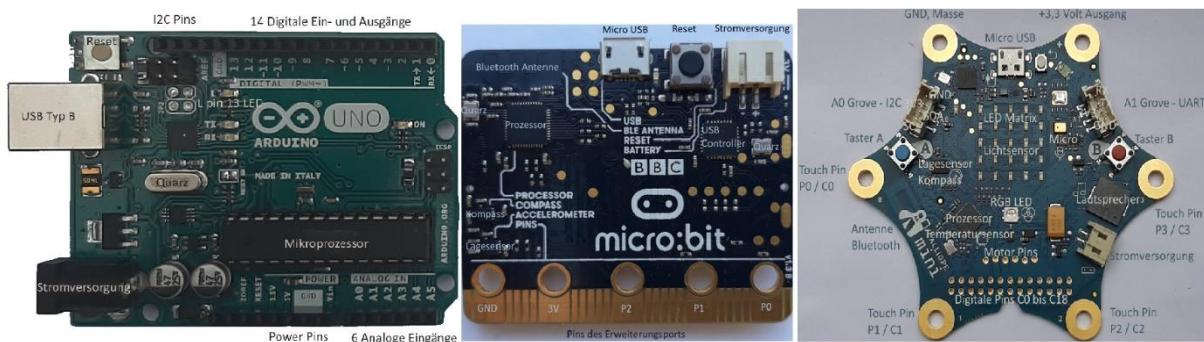


Abbildung 2 - Die Mikrokontroller Arduino, Micro:bit und Calliope mini, von links nach rechts.
Quelle: Eigene Aufnahmen, CC-BY-SA

Breakoutboards/Shields. Diese erweitern die Funktionalität der Mikrokontroller. Für den Arduino, welcher ja keine Sensoren besitzt, werden sogenannte Shields aufgesteckt. Für die folgenden Projekte sind ein Motor Shield und ein Grove Shield empfehlenswert.

Der Micro:bit kann auf Breakoutboards gesteckt werden, welche Motortreiber oder aber auch Grove Anschlüsse haben können, siehe Abbildung 3 und Abbildung 4.

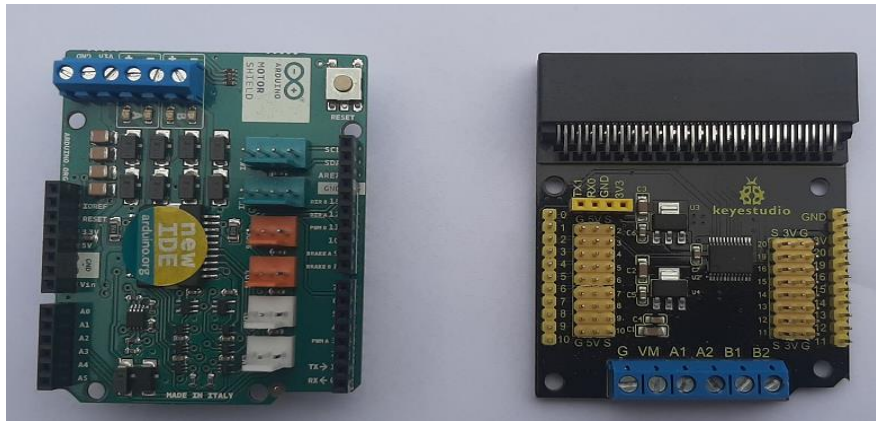


Abbildung 3- Motorshield für den Arduino (links) und Breakoutboard mit Motortreiber für den Micro:bit (rechts). Quelle: Eigene Aufnahme, CC-BY-SA

Grove Sensoren. Grove Sensoren müssen nicht mit den einzelnen Pins der Mikrokontroller verbunden werden, sondern mit einer Grove Verbindung. Der Calliope mini besitzt zwei Grove Buchsen, für den Arduino gibt es ein Grove Shield und für den Micro:bit ebenfalls ein Breakoutboard mit einem Grove Anschluss. Für Schülerinnen und Schüler mit geringen Elektronikkenntnissen empfiehlt es sich Grove-Sensoren zu verwenden. Grove ist lediglich der Typ der Buchse und kann intern entweder an einem I2C Bus oder an eine serielle Schnittstelle, der UART, angeschlossen sein. Aber auch ein Anschluss an einen Pin ist mit Grove möglich, zu sehen in Abbildung 3 bei dem Arduino Shield und dem Micro:bit Breakoutboard. Der Calliope mini besitzt mit der linken Grove Buchse einen Anschluss an den I2C Bus und mit der rechten einen Anschluss an die serielle Schnittstelle, siehe Abbildung 4. Die Hersteller der Grove-Sensoren legen noch ein Datenblatt mit näheren Informationen, bei aus dem ersichtlich ist, an welche Schnittstelle der Grove Sensor anzuschließen ist.

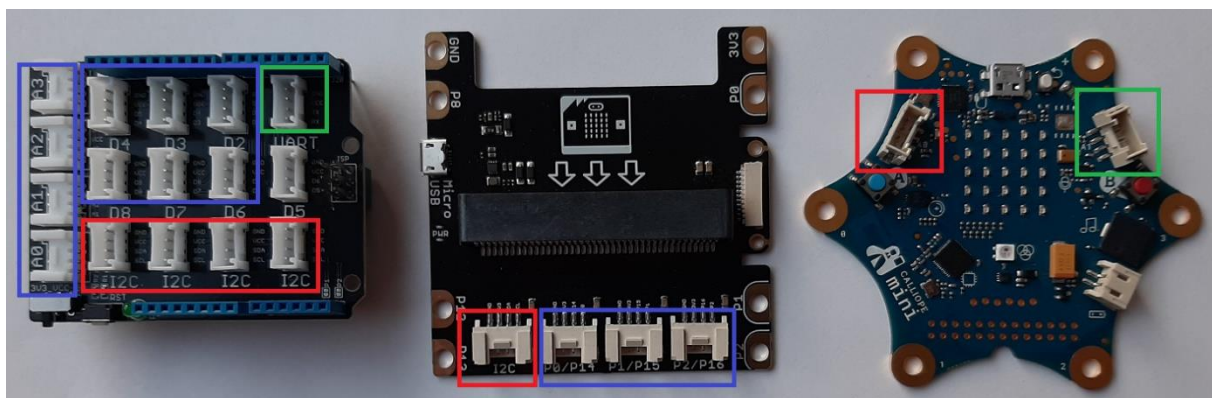


Abbildung 4 - Arduino Shield, Micro:bit Breakoutboard, Calliope mini. Die I2C Anschlüsse sind rot, die seriellen Anschlüsse (UART) grün und die Anschlüsse zu den Pins sind blau markiert, von links nach rechts. Quelle: Eigene Aufnahme, CC-BY-SA

5 Projekte

Die vorgestellte Auswahl von Projekten soll vor allem die Grunderfahrungen des Informatikunterrichts ermöglichen. Wegen der Projektcharakteristik ist nicht für alle Schülerinnen und Schüler gesichert, dass die gleichen Tätigkeiten ausgeführt werden können und daher ist es nicht möglich, Kompetenzen zu nennen, welche für alle gültig sein sollen sie zu erreichen. Im Gegensatz zu Aufgaben erscheint es auch wenig sinnvoll eine genaue Zeitangabe für die Fertigstellung eines Projekts anzugeben, da diese primär von der Anzahl der Teammitglieder abhängt, welche sich die Tätigkeiten aufteilen.

Ganz wesentlich im Sinne von Design Thinking ist die freie Wahlmöglichkeit der Mikrokontroller, damit einher gehen soll eine vertiefende Auseinandersetzung mit den Eigenschaften der zur Verfügung gestellten Systeme.

Anmerkungen

Die Eingänge der Pins sind grundsätzlich hochohmig, dadurch ist der Spannungspegel weder von 0 V noch von 3,3 V gewährleistet. Um einen Kontakt eindeutig zu registrieren, ist ein Pull-up Widerstand notwendig, welcher den Eingang des Mikrokontrollers auf +3,3V festlegt. Wird ein Kontakt mit der Masse hergestellt wird das Potential auf 0V heruntergezogen. Somit sind genau zwei Kontaktzustände möglich, welche vom Mikrokontroller registriert werden können.

Beim Calliope mini sind die herausgeführten Pins 0, 1, 2 und 3 bereits Touch-Pins, beim Micro:bit die Pins 0, 1 und 2, sie benötigen daher keine Pull-up Widerstände mehr. Anzumerken ist dabei, dass es sich nicht um kapazitive Touch-Pins handelt, es muss deshalb mit einer Hand immer Ground berührt werden und mit der anderen Hand der Pin.

Ein Vorschlag für die Erstellung einer Roboter Chassis oder anderer stabiler Plattformen ist die Verwendung von zwei Stück Pappe, welche gegeneinander um 90° angeordnet sind. Diese Vorgehensweise hat sich bei der Erstellung von Prototypen bewährt, da die Teile sehr stabil werden und sich trotzdem mit einer Schere zuschneiden lassen.

Die Projekte werden immer ohne der ohnehin notwendigen Mikrokontroller, USB-Kabel, Computer und Netzwerkverbindungen zu den Entwicklungs-Plattformen beschrieben.

5.1 Hot Wire

Story

Hot Wire bzw. Der heiße Draht ist ein Geschicklichkeitsspiel, bei dem mit einer auf einem Stab befestigten, leitfähigen Schlaufe einem Parcours, welcher aus einem gebogenen Draht besteht, gefolgt werden muss. Ein eventueller Kontakt von Stab und Parcours wird registriert und als Lichtsignal und Lautsignal angezeigt.

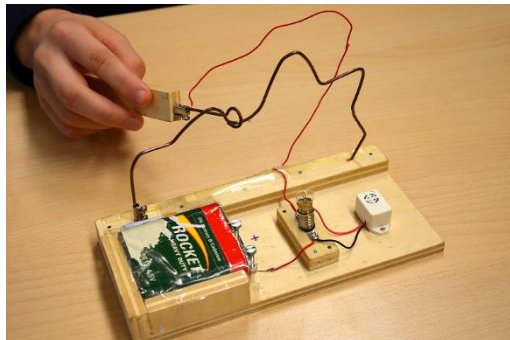


Abbildung 5 - Hot Wire ohne einem Mikrokontroller gefertigt.

Quelle: Schwabe, Bernd: "Heißer Draht ...", <https://commons.wikimedia.org>, CC-BY-SA

Organisation

Lehrerrolle	Scrum Master bei strukturierten Projekten Product Owner bei angeleiteten Projekten
Teamgröße	2 – 4 Schülerinnen und Schüler, optional mit Pair Programming
Kanbanboard	To-Do, Doing, Done mit vorbereitetem Sprintbacklog für strukturierte Projekte. To-Do, Next, Doing, Done mit vorbereitetem Product Backlog für angeleitete Projekte.
Story Sheet	https://www.tutory.de/dokument/487db406

Werkzeug

	Scheren, Abisolierzange, Seitenschneider
	Cuttermesser, Lineale

Material	
Mechanik	Pappe, Klebstoff
Kabel	Drahtbrücken-Stecker – bunt und verschiedene Längen Kabel mit Krokodil-Klemmen Lüsterklemmen
Elektronik	eine Rolle Aderleitung H07V-U 1,5 mm ² Widerstände: 100 W oder 220 W für die Pull-Up Widerstände, ½ Watt, zumindest ¼ Watt, sie haben stabilere Anschlussdrähte

Produkt-Backlog	
	Das Spiel muss stabil sein, von alleine stehen und transportierbar sein
	Das Spiel soll aus einem Parcours und einem Spielstab mit Schlinge bestehen
	Bei einem Kontakt des Spielstabs mit dem Parcours soll ein Lichtsignal gegeben werden
	Bei einem Kontakt soll zusätzlich ein Lautsignal erfolgen
	Der Parcours soll biegsam sein, um Spielvariationen zu ermöglichen
	Ein Mikrokontroller soll das Spielgeschehen steuern
	Optional können noch eine Starttaste und eine Stoptaste eingesetzt werden
	Optional kann noch die Anzahl der Kontakte bei einem Durchgang gezählt werden

Sprint-Backlog	
	Erkundigungen über bestehende Produkte und vorhandene Designs einholen
	Minimalste Funktionalität für einen Prototyp festlegen (was kann weggelassen werden)
	Anfertigen einer Skizze für einen Prototyp mit allen Bauteilen
	Suchen nach alternativen Lösungen
	Eine Liste mit den notwendigen Bauteilen anfertigen und die Bauteile besorgen
	Schaltplan mit Anschlüssen an den Mikrokontroller anfertigen
	Bestimmen der für die Funktion erforderlichen Ein- und Ausgänge, sowie der Sensoren und Anzeigen eines Mikrokontrollers
	Vergleichen der Mikrokontroller in Hinblick auf vorhandene Ein- und Ausgänge, sowie der Sensoren und Anzeigen
	Erkundigungen über die Funktionsweise und Realisierung eines Resistance-Touch-Pins einholen
	Die Funktionsweise des Programms, Schritt für Schritt und Zeile für Zeile, in Worten aufschreiben
	Ein Programm mit einer Testkonfiguration erstellen, das heißt alleine mit dem Mikrokontroller
	Einen Prototyp anfertigen
	Die Programmierung eines Touch-Pins erkunden
	Eine Fehleranalyse durchführen


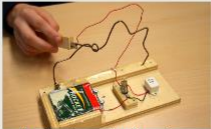
Lösungsmöglichkeit – Hot Wire

Story Sheet - Hot Wire

Geschicklichkeitsspiel - Der heiße Draht

Aufwand: **

Dieses Geschicklichkeitsspiel kann in verschiedenen Schwierigkeitsstufen gespielt werden, je nachdem wie der Draht gebogen wird. Auf dem Bild ist eine Version ohne der Verwendung eines Mikrokontrollers zu sehen.

Quelle: Schwabe, Bernd: "Heißer Draht...", <https://commons.wikimedia.org>, CC-BY-SA

Das Spiel besteht aus einem langen gebogenen Draht, Parcours genannt und einem Stab an dessen Ende sich eine Ring, bzw. Schlaufe befindet. Mit dem Stab soll die Schlaufe entlang des Parcours geführt werden ohne ihn zu berühren. Beide Teile müssen aus einem metallischen Material bestehen damit der Kontakt über einen Stromfluss registriert werden kann.

Anforderungen:

- Das Spiel muss stabil sein, von alleine stehen und transportierbar sein.
- Bei einem Kontakt soll der Mikrokontroller ein Lichtsignal geben.
- Bei einem Kontakt soll der Mikrokontroller auch noch ein Lautsignal geben.
- Der Parcours sollte biegsam sein.
- Das Spiel soll von einem Mikroprozessor gesteuert werden.
- Zusätzlich kann noch eine Start- und Stop-Taste verwendet werden.

Gestaltungshinweis:

- Werden zwei Lagen Pappe so aufeinander geklebt, dass die Rillen normal aufeinander stehen, bewirkt das eine stabile Basis.

Andreas Kämer, Private Pädagogische Hochschule der Diözese Linz

Informatik

Abbildung 7 - Story Sheet Hot Wire.
Quelle: Eigene Darstellung,
<https://www.tutory.de/dokument/487db406>

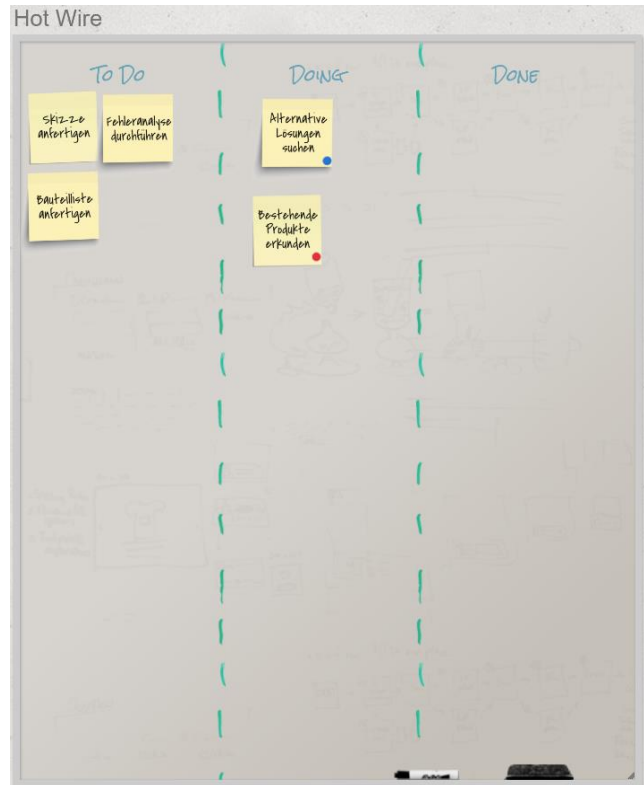


Abbildung 6 – Kanbanboard.
Quelle: Eigene Darstellung, <http://scrumbler.ca/hotwire>,
CC-BY-SA

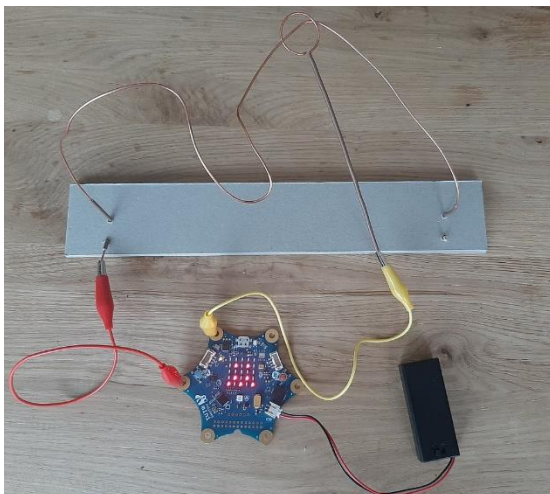


Abbildung 9 - Eine mögliche Realisierung von Hot Wire. Quelle: Eigene Aufnahme, CC-BY-SA

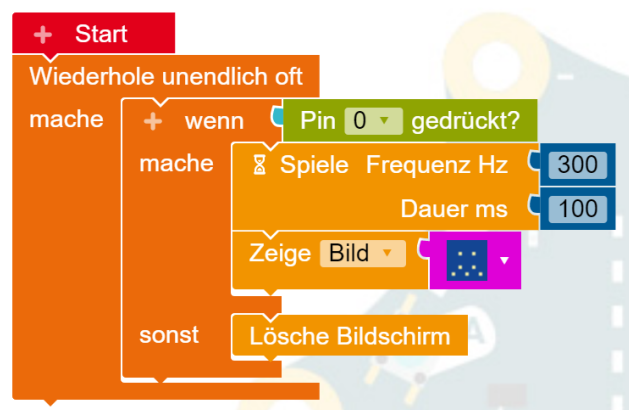


Abbildung 8 - Eine mögliche Realisierung des Programms in OpenRoport. Quelle: Eigene Darstellung, CC-BY-SA

5.2 Mars Rover

Story

Ein Mars Rover erkundet fremde Planeten und ist so weit entfernt, dass er nicht ferngesteuert werden kann, sondern völlig autonom fahren muss. Um nicht gegen Felsen zu stoßen benötigt der Rover Sensoren, welche Hindernisse erkennen können, um daraufhin die Fahrtrichtung zu ändern.



Abbildung 10 – Mars Rover. Quelle: Hugoviv, „Mars Rover“, <https://commons.wikimedia.org>, CC-BY-SA

Organisation

Lehrerrolle	Scrum Master bei strukturierten Projekten Product Owner bei angeleiteten Projekten
Teamgröße	2 – 4 Schülerinnen und Schüler, optional mit Pair Programming
Kanbanboard	To-Do, Doing, Done mit vorbereitetem Sprintbacklog für strukturierte Projekte. To-Do, Next, Doing, Done mit vorbereitetem Product Backlog für angeleitete Projekte.
Story Sheet	https://www.tutory.de/dokument/38d4e90b

Material	
Mechanik	Pappe, Klebstoff, kleine Schrauben
Kabel, Stecker	Drahtbrücken-Stecker – bunt und verschiedene Längen Kabel mit Krokodil-Klemmen; Buchsenleiste, 6-polig und Gummiband für Calliope Motoranschlüsse; Grove Kabel
Elektronik	Breakoutboard mit Motortreiber (Micro:bit); Motorshield (Arduino); Getriebemotoren, rechtwinkelig; Antriebsräder passend zum Getriebemotor; Ultraschallsensor, HC-SR04; Grove Ultraschallmodul, GRV Ultrasonic HC-SR04; Mikroschalter mit langem Metallhebel; 4,5V oder 9V-Batterie; Widerstände: 100 W oder 220 W für die Pull-Up Widerstände, falls Tastsensoren verwendet werden. ½ Watt, zumindest ¼ Watt, sie haben stabilere Anschlussdrähte

Produkt-Backlog	
	Der Rover soll eine maximale Größe von 20 x 20 cm haben
	Er soll sich nur mit zwei Rädern, welche auf rechtwinkelige Getriebemotoren gesteckt sind, fortbewegen
	Der Rover soll mindestens einen Sensor besitzen, um Hindernisse erkennen zu können
	Wird ein Hindernis erkannt so soll er die Fahrtrichtung ändern

Sprint-Backlog	
	Erkundigungen über bestehende Produkte und vorhandene Designs einholen
	Minimalste Funktionalität für einen Prototyp festlegen (was kann weggelassen werden)
	Anfertigen einer Skizze für einen Prototyp mit allen Bauteilen
	Suchen nach alternativen Lösungen
	Eine Liste mit den notwendigen Bauteilen anfertigen und Bauteile beschaffen
	Einen Schaltplan mit Anschlüssen an den Mikrokontroller anfertigen
	Bestimmen der für die Funktion erforderlichen Ein- und Ausgänge, sowie der Sensoren und Anzeigen eines Mikrokontrollers
	Vergleiche die Mikrokontroller in Hinblick auf vorhandene Ein- und Ausgänge, sowie der Sensoren und Anzeigen
	Erkundigungen über die Funktionsweise und Realisierung eines Resistance-Touch-Pins einholen
	Die Funktionsweise des Programms, Schritt für Schritt und Zeile für Zeile, in Worten aufschreiben
	Ein Programm mit einer Testkonfiguration erstellen, das heißt allein nur mit dem Mikrokontroller
	Einen Prototyp anfertigen
	Informationen zur Programmierung und zum Anschluss eines Ultraschallsensors einholen
	Informationen zur Programmierung und zum Anschluss eines Touch-Pins einholen
	Eine Fehleranalyse durchführen



Lösungsmöglichkeit – Mars Rover

Story Sheet - Mars Rover

Ein autonom fahrender Rover

Aufwand: ***

Ein Mars Rover erkundet einen fremden Planeten und ist so weit entfernt, dass er nicht ferngesteuert werden kann, sondern völlig autonom fahren muss. Um nicht gegen Felsen zu stoßen benötigt der Rover Sensoren, welche Hindernisse erkennen können, um daraufhin die Fahrtrichtung zu ändern.

Quelle:
https://commons.wikimedia.org/wiki/File:Marsovermsrds_simulation.jpg

Der Rover soll lediglich zwei Räder und am Heck eine Stütze besitzen. An der Frontseite des Rover soll ein Sensor, möglichst Tastsensoren oder ein Ultraschallsensor, erkennen ob ein Hindernis den Weg versperrt. Ist ein Hindernis erkannt, soll der Rover die Fahrtrichtung ändern.

Anforderungen:

- Der Rover soll eine maximale Größe von 20 x 20 cm haben.
- Er soll sich mit zwei Rädern, auf Gleichstrommotoren gesteckt, fortbewegen.
- Er soll mindestens einen Sensor besitzen, Tastsensor oder Ultraschallsensor

Gestaltungshinweis:

- Ein stabiles Chassis kann dadurch erreicht werden, dass zwei Lagen Wellpappe um 90° verdreht aufeinander geklebt werden.

Andreas Kiener, Private Pädagogische Hochschule der Diözese Linz

Informatick

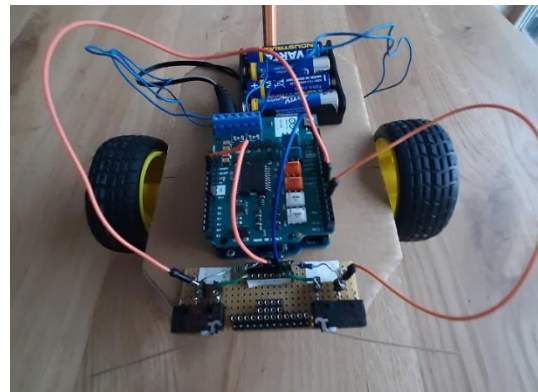
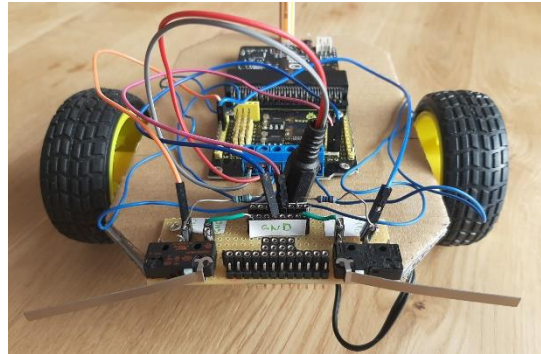
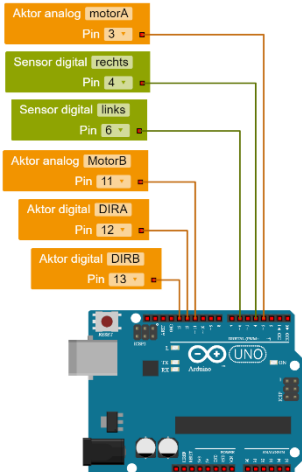


Abbildung 12 - Story Sheet Mars Rover. Quelle: Eigene Darstellung, <https://www.tutory.de/dokument/38d4e90b>, CC-BY-SA

Abbildung 11 - Mögliche Realisierung mit Micro:bit (oben) und Arduino (unten). Quelle: Eigene Aufnahmen, CC-BY-SA



```

+ Start
Wiederhole unendlich oft
mache
+ wenn gib digitalen Wert Sensor links = 0
mache
slowbwd
Warte ms 2000
Schreibe analogen Wert Aktor motorA = 100
Schreibe analogen Wert Aktor MotorB = 0
Warte ms 2000
sonst fastbwd
+ wenn gib digitalen Wert Sensor rechts = 0
mache
slowbwd
Warte ms 2000
Schreibe analogen Wert Aktor motorA = 0
Schreibe analogen Wert Aktor MotorB = 100
Warte ms 2000
sonst fastbwd
+ slowbwd
Schreibe digitalen Wert Aktor DIRA = 1
Schreibe digitalen Wert Aktor DIRB = 1
Schreibe analogen Wert Aktor motorA = 100
Schreibe analogen Wert Aktor MotorB = 100
+ fastbwd
Schreibe digitalen Wert Aktor DIRA = 0
Schreibe digitalen Wert Aktor DIRB = 0
Schreibe analogen Wert Aktor motorA = 255
Schreibe analogen Wert Aktor MotorB = 255
    
```

Abbildung 13 - Eine mögliche Realisierung des Programms für den Arduino mit Open Roberta. Quelle: Eigene Darstellung, CC-BY-SA

5.3 Anemometer

Story

Wetterstationen sind auf der ganzen Welt verteilt und messen vollautomatisch Wetterdaten, darunter auch die Windgeschwindigkeit. Diese Daten sind wichtig für Wettervorhersagen und für Vorhersagen zur Klimaveränderung. Das Anemometer, welches die Windgeschwindigkeit misst, ist im Bild zu sehen.



Abbildung 14 – Anemometer (rechts). Quelle: Farmatin, "Eureka Airport anemometer", <https://commons.wikimedia.org>, CC-BY-SA

Gestaltungshinweise:

Lüfter können auch als Generatoren verwendet werden und geben dabei eine der Drehzahl analoge Ausgangsspannung aus, welche verwendet werden kann, um die Windgeschwindigkeit zu ermitteln. Sie haben aber auch ein Anlaufmoment zu überwinden, wodurch es bei kleinen Windgeschwindigkeiten zu Ungenauigkeiten kommt.

Lüfter können so weit unter eine Betriebsspannung gesetzt werden, dass sie sich gerade nicht bewegen und können dann selbst bei kleinster Luftbewegung loslaufen.

Organisation

Lehrerrolle	Scrum Master bei strukturierten Projekten Product Owner bei angeleiteten Projekten
Teamgröße	2 – 4 Schülerinnen und Schüler, optional mit Pair Programming
Kanbanboard	To-Do, Doing, Done mit vorbereitetem Sprintbacklog für strukturierte Projekte. To-Do, Next, Doing, Done mit vorbereitetem Product Backlog für angeleitete Projekte.
Story Sheet	https://www.tutory.de/dokument/285389f1

Material	
Mechanik	Pappe; Klebstoff; kleine Schrauben; Autogenschweißdrähte 1,5 x 333 mm; Tischtennisbälle; Papp-Rollen;
Kabel, Stecker	Drahtbrücken-Stecker – bunt und verschiedene Längen Kabel mit Krokodil-Klemmen; 6polige Buchsenleiste; Gummiband für Calliopes Motoranschlüsse; Grove Kabel
Elektronik	Breakoutboard mit Motortreiber (Micro:bit); Motorshield (Arduino); Groveshield (Arduino); Ultraschallsensor, HC-SR04; Grove Ultraschallmodul, GRV Ultrasonic HC-SR04; Axiallüfter; 4,5V oder 9V-Batterie

Produkt-Backlog	
	Der Durchmesser des Windrades sollte ca. 20 cm betragen
	Der Lüfter darf soweit verändert bzw. zerstört werden, dass er den Anforderungen genügt. (Bohrungen, Ausbrechen von Lüfterschaufeln usw.)
	Das Messgerät soll mit einem Sensor die Umlaufzeit bestimmen
	Die Ausgabe der Windgeschwindigkeit kann auf einem 5x5 LED-Array erfolgen.
	Die Windgeschwindigkeit soll in m/s angezeigt werden

Sprint-Backlog	
	Erkundigungen über bestehende Produkte und vorhandene Designs einholen
	Minimalste Funktionalität für einen Prototyp festlegen (was kann weggelassen werden)
	Anfertigen einer Skizze für einen Prototyp mit allen Bauteilen
	Suchen nach alternativen Lösungen
	Eine Liste mit den notwendigen Bauteilen anfertigen und Bauteile beschaffen
	Einen Schaltplan mit Anschlüssen an den Mikrokontroller anfertigen
	Bestimmen der für die Funktion erforderlichen Ein- und Ausgänge, sowie Sensoren und Anzeigen eines Mikrokontrollers
	Vergleichen der Mikrokontroller in Hinblick auf vorhandene Ein- und Ausgänge, sowie der Sensoren und Anzeigen
	Erkundigungen über die Funktionsweise eines Lüfters einholen, Betriebsspannung, Drehzahl.
	Die Funktionsweise des Programms, Schritt für Schritt und Zeile für Zeile, in Worten aufschreiben
	Ein Programm mit einer Testkonfiguration erstellen, das heißt allein nur mit dem Mikrokontroller
	Einen Prototyp anfertigen
	Informationen zur Programmierung und zum Anschluss eines Ultraschallsensors einholen
	Informationen zur Programmierung einer Stoppuhr einholen
	Eine Fehleranalyse durchführen



Lösungsmöglichkeit – Anemometer

Story Sheet -Anemometer

Messung der Windgeschwindigkeit - Anemometer

Aufwand: ***

Wetterstationen sind auf der ganzen Welt verteilt und messen vollautomatisch Wetterdaten, darunter die Windgeschwindigkeit. Diese Daten sind wichtig für Wettervorhersagen und für Vorhersagen zur Klimaveränderung.

Anemometer, Quelle: https://commons.wikimedia.org/wiki/File:2014-10-03120038EurekaAirportAWOS_anemometer.JPG

Aus einer Drehbewegung, verursacht durch Wind, soll die Windgeschwindigkeit in m/s bestimmt werden. Es ist ein Prototyp, entsprechend der rechts in der Abbildung zu sehenden Apparatur, anzufertigen.

Die Windgeschwindigkeit lässt sich dadurch berechnen, dass die Zeit für eine Umdrehung gemessen wird. Der Kreisumfang, hervorgerufen durch die Drehbewegung, wird durch die gemessene Zeit dividiert.

Windgeschwindigkeit [m/s] = Umfang [m] / Umlaufzeit [s]

Gestaltungshinweis: Lüfter können auch als Generatoren verwendet werden und geben dabei eine der Drehzahl analoge Ausgangsspannung aus, welche verwendet werden kann, die Windgeschwindigkeit zu ermitteln. Sie haben aber auch ein Anlaufmoment zu überwinden, wodurch es bei kleinen Windgeschwindigkeiten zu Problemen kommt. Lüfter können so weit unter eine Betriebsspannung gesetzt werden, dass sie sich gerade nicht bewegen und werden bei kleinster Luftbewegung loslaufen.

Andreas Klierer, Private Pädagogische Hochschule der Diözese Linz

Informatik

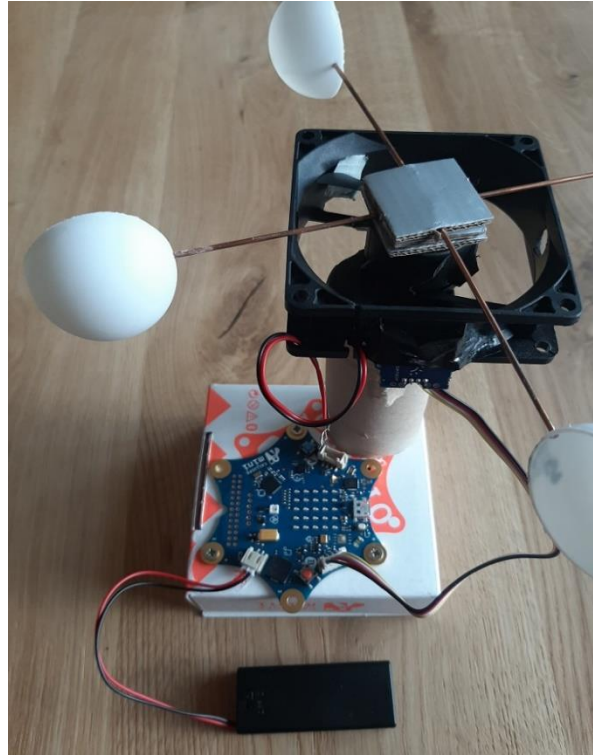


Abbildung 15 - Story Sheet Anemometer.
Quelle: Eigene Darstellung,
<https://www.tutory.de/dokument/285389f1>

Abbildung 16 - Eine mögliche Realisierung mit einem Lichtsensor aus dem Grove Starter Kit für Arduino, welcher die Anzahl der Umdrehungen pro Zeitintervall misst. Quelle: Eigene Darstellung, CC-BY-SA

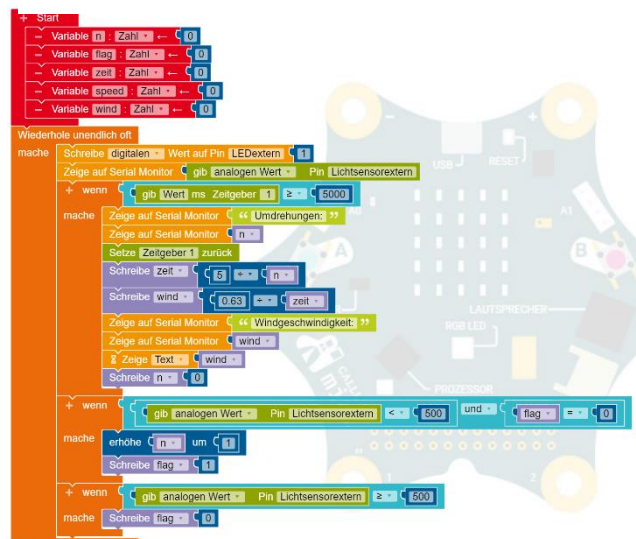
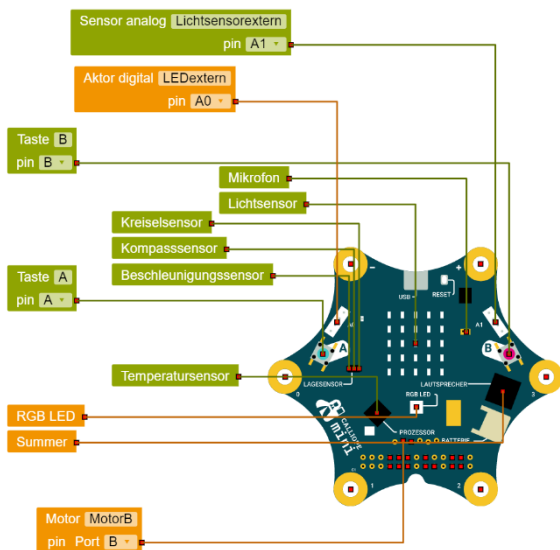


Abbildung 17 - Eine mögliche Realisierung des Programms zu dem oben abgebildeten Anemometer. Die Messdauer beträgt 5 sec und berechnet anhand der Anzahl der Umdrehungen die Zeit für eine Umdrehung. Quelle: Eigene Darstellung, CC-BY-SA

5.4 Anemoskop – Windrichtungsgeber

Story



Abbildung 18 - Anemoskop. Quelle: Farmatin, "Eureka Airport anemometer", <https://commons.wikimedia.org>, CC-BY-SA

Für eine Wettervorhersage ist neben der Windgeschwindigkeit auch noch die Windrichtung wesentlich. Es wird dabei jene Windrichtung angegeben, aus der der Wind kommt.

Gestaltungshinweise:

Das Anemoskop soll sich wie eine Fahne in den Wind drehen und zeigt mit dem Kopf in jene Richtung, aus der der Wind kommt. Für einen Prototyp ist es ausreichend nur vier Windrichtungen am LED-Array anzuzeigen: N ... Nord, S ... Süd, W...West und O ... Ost.

In nebenstehender Abbildung ist eine für das Projekt adaptierte Windrose zu sehen. Die angegebenen Winkel bezeichnen die Grenzen der Winkel zwischen den Himmelsrichtungen, welche der Kompass-Sensor ausgibt.

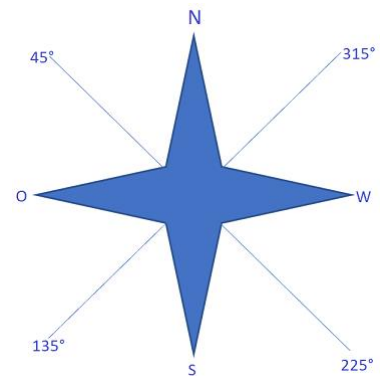


Abbildung 19 - Windrose. Quelle: Eigene Darstellung, CC-BY-SA

Organisation

Lehrerrolle	Scrum Master bei strukturierten Projekten Product Owner bei angeleiteten Projekten
Teamgröße	2 – 4 Schülerinnen und Schüler, optional mit Pair Programming
Kanbanboard	To-Do, Doing, Done mit vorbereitetem Sprintbacklog für strukturierte Projekte.

	To-Do, Next, Doing, Done mit vorbereitetem Product Backlog für angeleitete Projekte
Story Sheet	https://www.tutory.de/dokument/5b374214

Material	
Mechanik	Pappe; Papp-Rollen; Klebstoff; kleine Schrauben; Autogenschweißdrähte 1,5 x 333 mm
Elektronik	Axiallüfter

Produkt-Backlog	
	Der Höhe des Windgebers sollte 20 cm nicht überschreiten.
	Der Lüfter darf so verändert bzw. zerstört werden, dass er den Anforderungen genügt
	Die Anzeige der Himmelsrichtungen ist auf N, W, S und O beschränkt
	Die Ausgabe kann auf einem 5x5 LED-Array erfolgen
	Das Anemoskop soll wie eine Windfahne auf den einfallenden Wind reagieren

Sprint-Backlog	
	Erkundigungen über bestehende Produkte und vorhandene Designs einholen
	Minimalste Funktionalität für einen Prototyp festlegen (was kann weggelassen werden)
	Anfertigen einer Skizze für einen Prototyp mit allen Bauteilen
	Suchen nach alternativen Lösungen
	Eine Liste mit den notwendigen Bauteilen anfertigen und Bauteile beschaffen
	Erkundigung über die Funktionsweise und Kalibrierung eines Kompasses einholen
	Informationen zur Programmierung des Kompasses einholen
	Vergleichen der Mikrokontroller in Hinblick auf vorhandene Sensoren und Anzeigen
	Die Funktionsweise des Programms, Schritt für Schritt und Zeile für Zeile, in Worten aufschreiben
	Ein Programm mit einer Testkonfiguration erstellen, das heißt allein nur mit dem Mikrokontroller
	Einen Prototyp anfertigen
	Eine Fehleranalyse durchführen



Lösungsmöglichkeit – Windrichtungsgeber

Story Sheet - Anemoskop

Windrichtungsgeber - Anemoskop

Aufwand: ***


Für die Wettervorhersage sind neben der Windgeschwindigkeit auch noch die Windrichtung wichtig. Es wird jene Windrichtung angegeben aus der der Wind kommt.

Anemoskop, Quelle: https://commons.wikimedia.org/wiki/File:2014-10-03170038EurekaAirportAWOS_anemometer.JPG

Das Anemoskop soll sich wie eine Fahne in den Wind drehen und zeigt mit dem Kopf in jene Richtung aus der der Wind kommt. Für einen Prototyp ist es ausreichend nur vier Windrichtungen am LED-Array anzuzeigen: N...Nord, S...Süd, W...West und O...Ost.

In nebenstehender Abbildung ist eine für das Projekt adaptierte Windrose zu sehen. Die angegebenen Winkel bezeichnen die Grenzen der Winkel zwischen den Himmelsrichtungen, welcher der Kompass-Sensor ausgibt.



Andreas Kiener, Private Pädagogische Hochschule der Diözese Linz

Informatik

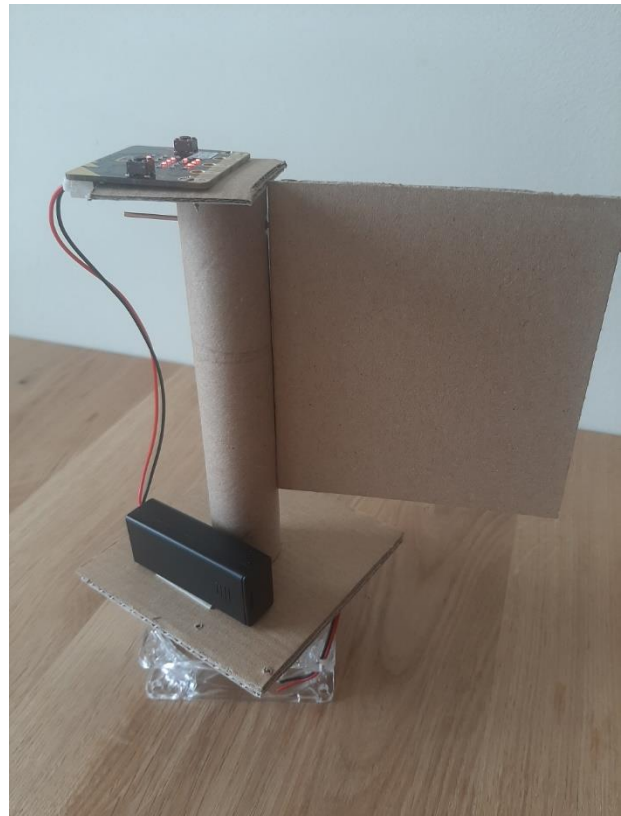


Abbildung 22 - Story Sheet - Windrichtungsgeber, Quelle: Eigene Darstellung, <https://www.tutory.de/dokument/5b374214>, CC-BY-SA

Abbildung 21 - Eine mögliche Realisierung eines Anemoscops. Quelle: Eigene Aufnahme, CC-BY-SA

Abbildung 20 - Eine mögliche Realisierung des Programms eines Anemoscops, erstellt auf: <https://makecode.microbit.org/> Quelle: Eigene Darstellung, CC-BY-SA

5.5 Schrittzähler

Story

Ein Schrittzähler ist Bestandteil aller Fitnesstracker, daraus lassen sich Informationen über den Kalorienverbrauch gewinnen. Beschleunigungssensoren können Erschütterungen, welche durch das Gehen verursacht werden, detektieren. Schrittzähler können in Kleidungsstücke eingearbeitet werden und zählen zu den Wearable Computing Anwendungen.

Es ist ein Schrittzähler zu bauen, welcher am Fußgelenk getragen werden kann. Die Anzeige kann auf der LED-Matrix eines der Mikrocontroller erfolgen.

Anforderungen:

Der Schrittzähler sollte über das Fußgelenk übergezogen werden.

Die Anzeige soll bei jedem Schritt aktualisiert werden

Mit einer Taste soll die Anzeige zurückgesetzt werden und ein neuer Zählvorgang beginnen.

Gestaltungshinweis:

Elastische Materialien und Stoffe eignen sich gut, um einen Mikrocontroller einerseits befestigen zu können, als auch am Fußgelenk genügend Halt zu bieten.



Abbildung 23 - Wearables.
Quelle: CSIRO, "CSIRO textile",
<https://commons.wikimedia.org/>, CC-BY-SA

Organisation	
Lehrerrolle	Option 1: Scrum Master für strukturierte Projekte Option 2: Product Owner für angeleitete Projekte
Teamgröße	2 Schülerinnen und Schüler
Kanbanboard	Option 1: To-Do, Doing, Done mit vorbereitetem Sprintbacklog Option 2: To-Do, Next, Doing, Done mit vorbereitetem Produktbacklog
Story Sheet	https://www.tutory.de/dokument/3a8c4083

Material	
Mechanik	Stoffe, Klettband, Zwirn
Werkzeug	Schere, Nähwerkzeug

Produkt-Backlog	
	Der Schrittzähler soll am Fußgelenk getragen werden
	Er soll nur die Schritte eines Fußes zählen
	Die Anzeige soll auf einem 5x5 LED-Array erfolgen
	Die Schrittzahl soll nach jedem Schritt aktualisiert und ausgegeben werden
	Mit einem Taster soll die Anzeige zurückgesetzt werden können

Sprint-Backlog	
	Erkundigungen über bestehende Produkte und vorhandene Designs einholen
	Minimalste Funktionalität für einen Prototyp festlegen (was kann weggelassen werden)
	Anfertigen einer Skizze für einen Prototyp mit allen Bauteilen
	Suchen nach alternativen Lösungen
	Eine Liste mit den notwendigen Bauteilen anfertigen und Bauteile beschaffen
	Vergleichen der Mikrokontroller in Hinblick auf vorhandene Sensoren und Anzeigen
	Erkundigungen über die Funktionsweise eines Beschleunigungssensors einholen
	Die Funktionsweise des Programms, Schritt für Schritt und Zeile für Zeile, in Worten aufschreiben
	Ein Programm mit einer Testkonfiguration erstellen, das heißt nur mit dem Mikrokontroller allein
	Einen Prototyp anfertigen
	Informationen zur Programmierung einer Stoppuhr einholen
	Eine Fehleranalyse durchführen

Lösungsmöglichkeit – Schrittzähler

Story Sheet - Schrittzähler

Wearable Computing - Schrittzähler

Schwierigkeitsgrad: *

Ein Schrittzähler ist Bestandteil aller Fitnesstracker, daraus lassen sich Informationen über den Kalorienverbrauch gewinnen. Beschleunigungssensoren können Erschütterungen, welche durch das Gehen verursacht werden, detektieren. Schrittzähler können in Kleidungsstücke eingearbeitet werden und zählen zu den Wearable Computing Anwendungen.

Es ist ein Schrittzähler zu bauen, welcher am Fußgelenk getragen werden kann. Die Anzeige kann auf der LED-Matrix eines der Microcontroller erfolgen.

Anforderungen:

- Der Schrittzähler sollte über das Hand- bzw. Fußgelenk übergezogen werden.
- Die Anzeige soll bei jedem Schritt aktualisiert werden
- Mit einer Taste soll die Anzeige zurückgesetzt werden und ein neuer Zählvorgang beginnen.

Gestaltungshinweis:

- Elastische Materialien und Stoffe eignen sich gut um einen Microcontroller einerseits zu befestigen zu können als auch am Handgelenk genügend Halt zu bieten.

Bildunterschrift/Quelle: <https://commons.wikimedia.org/wiki/File:CSIROScienceImage7664TheWearableBodyMappingSleeve.jpg>

Andreas Kiener, Private Pädagogische Hochschule der Diözese Linz

Informatik



Abbildung 25 - Schrittzähler. Quelle: Eigene Darstellung, <https://www.tutor.de/dokument/3a8c4083>, CC-BY-SA

Abbildung 26 - Mögliche Realisierung eines Schrittzählers. Quelle: Eigene Aufnahme, CC-BY-SA

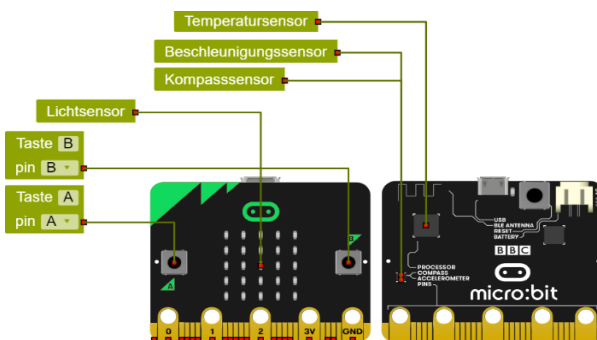


Abbildung 24 - Mögliche Realisierung des Programms zum Schrittzähler. Quelle: Eigene Darstellung, CC-BY-SA

Literaturverzeichnis

- [1] B. Bethge and M. Fothe, "Gründerfahrungen des Informatikunterrichts - ein Beitrag zur Frage der Allgemeinbildung von Informatik," in *INFOS 2013: 15. GI-Fachtagung "Informatik und Schule"* ; Praxisband ; 26.-28. September 2013 in Kiel, 2013, pp. 113–121.
- [2] B. Bethge and M. Fothe, "Gründerfahrungen im Informatikunterricht: Eine kompakte Beschreibung des Beitrags der informatischen Bildung für die Allgemeinbildung," *LOG IN*, 178/179, pp. 36–40, 2014.
- [3] K. Baum, N. Kirsch, K. Reese, P. Schmidt, L. Wachter, and V. Wolf, "Informatikunterricht in der Grundschule? – Erprobung und Auswertung eines Unterrichtsmoduls mit Calliope mini," (in de), *1617-5468*, 2019, doi: 10.18420/infos2019-b1.
- [4] A. Pasternak, Ed., *Informatik für alle*. Bonn: Gesellschaft für Informatik, 2019.
- [5] A. Bachinger and M. Teufel, *Computational Thinking mit BBC micro:bit: Digitale Bildung in der Sekundarstufe*, 2018.
- [6] L. W. Anderson and D. R. Krathwohl, *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. New York: Longman, 2001.
- [7] P. J. Denning, "Great Principles of Computing," *Communications of the ACM*, vol. 46, no. 11, pp. 15–20, 2003.
- [8] Y. Li *et al.*, "On Thinking and STEM Education," *Journal for STEM Educ Res*, vol. 2, no. 1, pp. 1–13, 2019, doi: 10.1007/s41979-019-00014-x.
- [9] J. M. Wing, "Computational thinking," *Communications of the ACM*, vol. 49, no. 3, pp. 33–35, 2006, doi: 10.1145/1118178.1118215.
- [10] A. V. Aho, "Ubiquity symposium: Computation and Computational Thinking," *Ubiquity*, vol. 2011, January, p. 313, 2011, doi: 10.1145/1922681.1922682.
- [11] P. J. Denning, "Beyond computational thinking," *Communications of the ACM*, vol. 52, no. 6, pp. 28–30, 2009, doi: 10.1145/1516046.1516054.
- [12] F. J. García-Peñalvo and J. Cruz-Benito, "Computational thinking in pre-university education," in *Proceedings TEEM 16*, pp. 13–17.
- [13] T. Lindberg, C. Meinel, and R. Wagner, "Design Thinking: A Fruitful Concept for IT Development?," in *Understanding Innovation, Design thinking: Understand - improve - apply*, H. Plattner, C. Meinel, and L. Leifer, Eds., Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2011, pp. 3–18.
- [14] Y. Doppelt, M. M. Mehalik, C. D. Schunn, E. Silk, and D. Krysinski, "Engagement and achievements: A case study of design-based learning in a science context.," *Journal of Technology Education*, vol. 19, no. 2, pp. 22–39, 2008.
- [15] D. Henriksen, "Creating STEAM with Design Thinking: Beyond STEM and Arts Integration," *The STEAM Journal*, vol. 3, no. 1, pp. 1–11, 2017, doi: 10.5642/steam.20170301.11.

- [16] S. M. Gomez Puente, M. van Eijck, and W. Jochems, "Empirical validation of characteristics of design-based learning in higher education," *International Journal of Engineering Education*, vol. 29, no. 2, pp. 491–503, 2013.
- [17] S. S. Gómez Puente, "Design-based learning : exploring an educational approach for engineering education," 2014.
- [18] M. Mehling, J. Schneider, T. Schmidt, and P. Reinelt, "Eine Wetterstation mit dem Calliope mini," Code your Life, 21st Century Competence Center im fjs e.V., Berlin, Apr. 2017. Accessed: Jan. 30 2021. [Online]. Available: <https://code-your-life.org/Mitmachen/mediabase/pdf/2780.pdf>
- [19] E. L. Deci and R. M. Ryan, "Self-determination theory: A macrotheory of human motivation, development, and health," *Canadian Psychology/Psychologie canadienne*, vol. 49, no. 3, pp. 182–185, 2008, doi: 10.1037/a0012801.
- [20] W. Klafki, "Selbstständiges Lernen muss gelernt werden!," in *Beiträge zur gymnasialen Oberstufe*, vol. 5, *Selbstständiges Lernen in der Schule: Selbstständiges Lernen in der Schule*, F. Stübiger and C. Schäfer, Eds., Kassel: Kassel Univ. Pr. GmbH, 2003, pp. 19–57.
- [21] D. Hänsel, *Projektunterricht: Ein praxisorientiertes Handbuch*, 2nd ed. Weinheim: Beltz, 1999.
- [22] K. Frey, *Die Projektmethode: »Der Weg zum bildenden Tun«*. s.l.: Beltz Verlagsgruppe, 2010.
- [23] P. Salza, P. Musmarra, and F. Ferrucci, "Agile Methodologies in Education: A Review," in *Agile and Lean Concepts for Teaching and Learning: Bringing Methodologies from Industry to the Classroom*, D. Parsons and K. MacCallum, Eds., Singapore: Springer Singapore, 2019, pp. 25–45.
- [24] R. Romeike and T. Göttel, "Agile projects in high school computing education," in *Proceedings of the 7th Workshop in Primary and Secondary Computing Education*, 2012, p. 48.
- [25] B. Gloger, *Scrum: Produkte zuverlässig und schnell entwickeln*, 5th ed. München: Hanser, 2016.
- [26] K. Schwaber and J. Sutherland, *The scrum guide*, 2020. [Online]. Available: <https://key2agile.de/wp-content/uploads/2020/11/2020-scrum-guide-german.pdf>
- [27] J. Sutherland, *The eduScrum guide*, 2015. Accessed: Jan. 14 2021. [Online]. Available: https://www.eduscrum.nl/img/The_eduScrum_guide_German_2.pdf
- [28] I. P. Artayasa, H. Susilo, U. Lestari, and S. E. Indriwati, "The effectiveness of the three levels of inquiry in improving teacher training students' science process skills," *Journal of Baltic Science Education*, 2017.
- [29] K. D. Simons and J. D. Klein, "The Impact of Scaffolding and Student Achievement Levels in a Problem-based Learning Environment," *Instr Sci*, vol. 35, no. 1, pp. 41–72, 2007, doi: 10.1007/s11251-006-9002-5.
- [30] H. Banchi and R. Bell, "The Many Levels of Inquiry," *Science and Children*, 2008.
- [31] J. Dinis-Carvalho, A. Ferreira, C. Barbosa, C. Lopes, H. Macedo, and P. Tereso, "Effectiveness of scrum in project based learning: Students view," in *Innovation, engineering and entrepreneurship*, 2018, pp. 1118–1124.
- [32] G. Braught, T. Wahls, and L. M. Eby, "The Case for Pair Programming in the Computer Science Classroom," *ACM Trans. Comput. Educ.*, vol. 11, no. 1, pp. 1–21, 2011, doi: 10.1145/1921607.1921609.

- [33] A. Kiener, "Physical Computing im Informatikunterricht-Eindrücke und Erfahrungen von Schülerinnen und Schülern," *LOG IN*, vol. 36, 185/186, pp. 54–59, 2016.
- [34] T. Schmalfeldt, "Einsatz von Skill Cards und Story Cards für einen kreativitätsfördernden Informatikunterricht auf der Sekundarstufe I," in *Informatik für alle*, A. Pasternak, Ed., Bonn: Gesellschaft für Informatik, 2019, pp. 305–314.
- [35] Fraunhofer IAIS, *OpenRoberta*. [Online]. Available: <https://lab.open-roberta.org/> (accessed: Jan. 13 2021).
- [36] Micro:bit Educational Foundation, *micro:bit Let's code*. [Online]. Available: <https://microbit.org/code/> (accessed: Jan. 30 2021).
- [37] C. Severance, "Massimo Banzi: Building Arduino," *Computer*, vol. 47, no. 1, pp. 11–12, 2014.

Abbildungsverzeichnis

Abbildung 1 - Kanbanboard eines fertig gestellten Projekts und ein Kanbanboard auf scrumblr.ca. .	10
Abbildung 2 - Die Mikrokontroller Arduino, Micro:bit und Calliope mini, von links nach rechts.	14
Abbildung 3- Motorshield für den Arduino (links) und Breakoutboard mit Motortreiber für den Micro:bit (rechts).....	15
Abbildung 4 - Arduino Shield, Micro:bit Breakout-Board, Calliope mini. Die I2C Anschlüsse sind rot, die seriellen Anschlüsse (UART) grün und die Anschlüsse zu den Pins sind blau markiert, von links nach rechts.	15
Abbildung 5 - Hot Wire ohne einem Mikrokontroller gefertigt.	17
Abbildung 6 – Kanbanboard.....	19
Abbildung 7 - Story Sheet Hot Wire.	19
Abbildung 8 - Eine mögliche Realisierung des Programms in OpenRoberta.	19
Abbildung 9 - Eine mögliche Realisierung von Hot Wire.....	19
Abbildung 10 – MarsRover.....	20
Abbildung 11 - Mögliche Realisierung mit Micro:bit (oben) und Arduino (unten).....	22
Abbildung 12 - Story Sheet Mars Rover.	22
Abbildung 13 - Eine mögliche Realisierung des Programms für den Arduino mit Open Roberta.	22
Abbildung 14 – Anemometer (rechts).....	23
Abbildung 15 - Story Sheet Anemometer.	25
Abbildung 16 - Eine mögliche Realisierung mit einem Lichtsensor aus dem Grove Starter Kit für Arduino, welcher die Anzahl der Umdrehungen pro Zeitintervall misst.	25
Abbildung 17 - Eine Mögliche Realisierung des Programms zu dem oben abgebildeten Anemometer. Die Messdauer beträgt 5 sec und berechnet anhand der Anzahl der Umdrehungen die Zeit für eine Umdrehung.	25
Abbildung 18 – Anemoskop.	26
Abbildung 19 – Windrose.....	26
Abbildung 20 - Eine mögliche Realisierung des Programms eines Anemoskops.....	28
Abbildung 21 - Eine mögliche Realisierung eines Anemoskops.	28

Abbildung 22 - Story Sheet - Windrichtungsgeber.....	28
Abbildung 23 - Wearables.....	29
Abbildung 26 - Mögliche Realisierung des Programms zum Schrittzähler.....	31
Abbildung 24 - Schrittzähler.....	31
Abbildung 25 - Mögliche Realisierung eines Schrittzählers.	31

Tabellenverzeichnis

Tabelle 1 - Gegenüberstellung der Great Principles of Computing von Denning zu Denkweisen und Methoden eines Informatikunterrichts.....	29
Tabelle 2 - Der iterative Prozess von Design Thinking	3
Tabelle 3 - Dimensionen und Charakteristika von Design-based Learning.....	4
Tabelle 4 - Rollenverteilung für Design-based Learning in Abhängigkeit der Strukturierung	9

Anhang

Story Sheets	
Hotwire	https://www.tutory.de/dokument/487db406
Mars Rover	https://www.tutory.de/dokument/38d4e90b
Anemometer	https://www.tutory.de/dokument/285389f1
Anemoskop	https://www.tutory.de/dokument/5b374214
Schrittzähler	https://www.tutory.de/dokument/3a8c4083

Skill Cards/Lernkarten zu Eigenschaften der Mikrokontroller	
Arduino	https://www.tutory.de/dokument/b3455865
Micro:bit	https://www.tutory.de/dokument/641c0b5e
Calliope mini	https://www.tutory.de/dokument/d095793b

Skill Cards/Lernkarten für den Verbindungsaufbau zu Arduino, Micro:bit und Calliope mini	
Arduino	https://www.tutory.de/dokument/7f0a123e
Micro:bit Calliope mini	https://www.tutory.de/dokument/0aad855b
OpenRoberta Connector	https://www.tutory.de/dokument/9c7ca2b7 https://www.tutory.de/dokument/10e6a2f9 https://www.tutory.de/dokument/29efd2dd

Skill Cards und dafür geeignete Unterlagen für den Calliope mini	
PHSZ	https://blogs.phsg.ch/robotik/files/2020/08/Calliope_Challenge_Cards_V2.4_Druckvorlage.pdf
RWT Aachen	http://schuelerlabor.informatik.rwth-aachen.de/materialien
Börding, Josef	https://jira.iais.fraunhofer.de/wiki/display/ORInfo/Gewusst+wie+...+Calliope
APPCAMPS	https://appcamps.de/unterrichtsmaterial/calliope-mini/

Für den Arduino als Lernkarten geeignete Unterlagen	
Math-Activity Center	https://math-activity.center/sites/default/files/material/SkriptArduino.pdf
RWT Aachen	http://schuelerlabor.informatik.rwth-aachen.de/materialien?page=3

Skill Cards und dafür geeignete Unterlagen für den Micro:bit	
Microbit.org	https://makecode.microbit.org/examples
Digital technologies hub	https://www.digitaltechnologieshub.edu.au/docs/default-source/webinar-slides-and-handouts/handout-2_-bbc-microbit.pdf
RWT Aachen	http://schuelerlabor.informatik.rwth-aachen.de/materialien

Weitere Entwicklungsumgebungen	
Microbit.org	https://makecode.microbit.org/
Arduino	http://technologiescollege.github.io/Blockly-at-rduino/
Arduino Grove Blocks	http://blocklyduino.github.io/BlocklyDuino/blockly/apps/blocklyduino/