Strichcode Lesegerät

Aufgabenstellung:

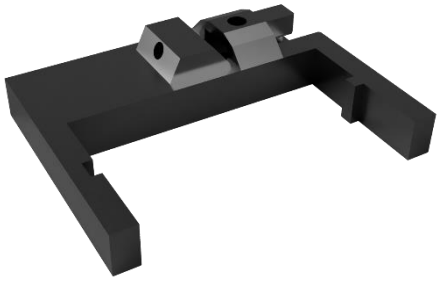
Die Grundidee dieses „kleinen“ Projektes war es mit Hilfe einer LED und eines Phototransistors einen Barcode auszulesen, interpretieren und auszugeben.

Die Arbeit des Projektes besteht also aus einer Hardwarekomponente, wie platziere ich die beiden Sensoren in Relation zueinander am besten um den höchsten Kontrast zwischen weis und schwarz zu erreichen und der Software, welche uns das eingelesene Signal speichert und daraus die Zahlenfolge entziffern kann.

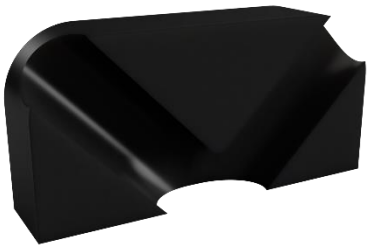
Lösung der Hardware

Erste Idee der Hardware

Bei der Hardwarefrage wurde sich nach anfänglichen Überlegungen mit einem Prisma zu arbeiten um das Licht am effizientesten um eine Ecke zu bekommen. Dies stellte sich schnell als ein zu großer Aufwand dar, da dadurch einige neue Probleme entstanden. (Rechts zu erkennen schnelles Design mit Halterungen für die LED/Sensor Steckbrett und Prisma/Linse)



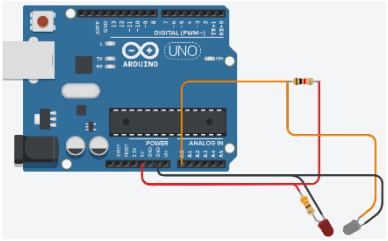
Zweite Idee der Hardware



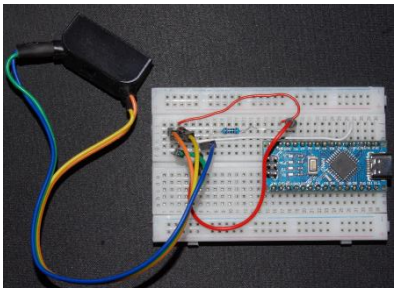
Diese Idee wurde nach einem ersten Prototyp wieder aufgegeben und durch eine viel simplere Variante ersetzt. Diese besteht lediglich aus einem Block mit zwei im Lot zueinander verlaufenden Röhren, in welche jeweils LED und Phototransistor passen. Die beiden Röhren wurde genauso verschoben, dass sich in der LED jeweils gegenüberliegenden, möglichst viel Lichtreflektion erkennen lässt. Diese Variante lieferte sehr zufriedenstellende Ergebnisse beim Auswerten der Analogen Signale. Von ca. 600 bis 1000 bei Analoger Auswertung mit 10-Bit (0-1023).

Aufbau

Weiters brauchen wir auser die schon erwähnten Komponenten wie der LED und dem Phototransistor noch jeweils einen Widerstand, Kabel Schrumpfschlauch etc., um die Daten zu verarbeiten brauchen wir einen Mikroprozessor in diesem Fall ein Arduino Nano und ein Steckbrett.



Links zu sehen ist einmal der Schematische Aufbau der Schaltung, für die LED wurde ein widerstand von 330Ω Widerstand verwendet und als zweiter Teil des Spannungsteilers mit dem Phototransistor wurde ein 1k Widerstand verwendet und darunter der Schlussendliche Aufbau.



Kostenberechnung und Arbeitszeiten

Insgesamt wurden an diesem Projekt sechs Wochen gearbeitet vier davon war es möglich in den Schulstunden zur Arbeit zu verwenden die restlich zwei Wochen setzen sich durch Ausfällen von Schulstunden und Ferien zusammen.

Woche	1	2	3	4	5	6
Stunden (+Außerschulische Stunden)	2	4 +1	2 +2	4	2 +1	0 +3
Leistung	Besprechung des Projektes Aufgabenstellung und Grundprinzipien des Sensors	Theoretischer Aufbau und Planung der Schaltung. Aufbau auf Steckbrett	Testen von verschiedenen Methoden LED und Phototransistor zu positionieren	Festgelegt auf die Befestigung durch eine 3D-Gedruckte Halterung	Begin mit dem Programmieren erste analoge auslesung und „Übersetzung“ in ein Binäres Signal	Übersetzung der digitalen Daten in das Dezimlasystem

Bauteil	***Preis***
Arduino Nano	≈ 3€
LED	≈ 0,5€
Phototransistor	≈ 0,5€
Widerstände	≈ 0,5€
3D Druck	≈ 0,5€

Ausarbeitung des Codes

Der Grundgedanke bei der Ausarbeitung des Codes war, dass bei jeder Flanke (Farbenänderung) die Zeit gespeichert wird nach dem Abscannen das Ergebnis ermittelt und ausgegeben wird.

Zum brechenden des Ergebnisses wird das Standardisierte „European Article Number„System verwendet in diesem Fall das EAN-8 System. Es besteht aus 8 Nummern, sieben für die Identifizierung und einer Checksum.

Um die gespeicherten Zeiten zu verwenden, machen wir uns den Anfang des Codes zu nutzen, da dieser immer aus drei gleich langen Str9ichen besteht und uns somit dreimal eine Zeit liefert, aus der wir einen Mittelwert errechnen können, mit dem wir später die Länge der einzelnen Flanken feststellen können. (gleiches tritt in der Mitte sowie am Ende des Codes auf)

Innerhalb des Barcodes werden die Zahlen mit einem Binärem System dargestellt wobei jeder Zahl 7 Bit zugeordnet werden. So schaut also der oben geschriebene Barcode Binärer ausgeschrieben so aus. Ein weiters wichtiges Detail ist hierbei, dass ein solcher code aus 67 Bit besteht.

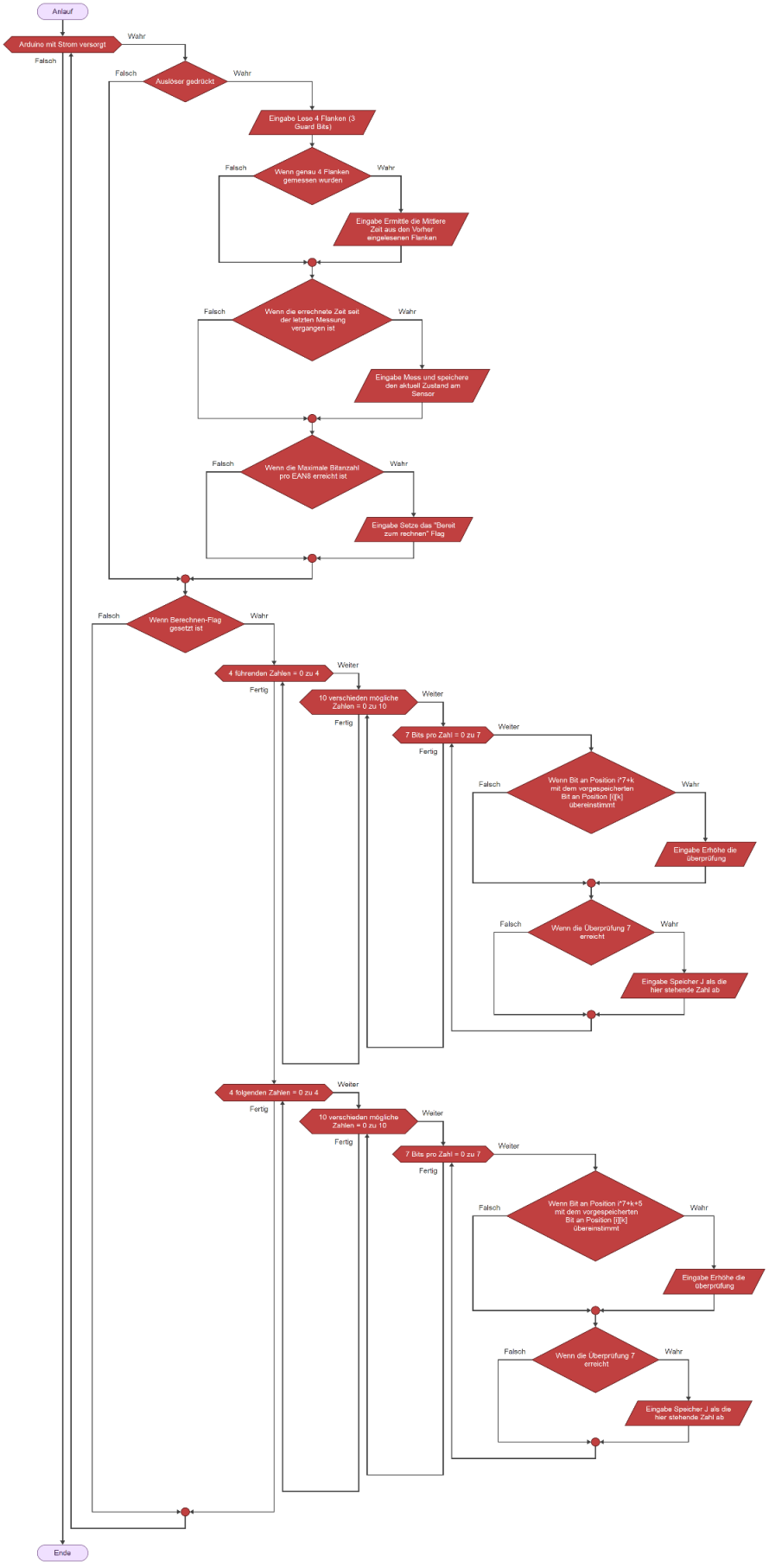
||	1	2	3	4	||	5	6	7	0	||
101	0011001	0010011	0111101	0100011	01010	1011100	1010000	1000100	1110010	101

Da hier ein eigenes Binärsystem verwendet wird, muss auch darauf Acht gegeben werden, vor allem da sich die Darstellung der Zahlen Jeh nach Position verändert, abhängig davon, ob sie vor oder nach der Mittleren „Guard Zone“ steht. Um diese Zahlen zu erlangen kann die folgende Tabelle verwendet werden.

Nummer	Links	Rechts
0	0001101	1110010
1	0011001	1100110
2	0010011	1101100
3	0111101	1000010
4	0100011	1011100
5	0110001	1001110
6	0101111	1010000
7	0111011	1000100
8	0110111	1001000
9	0001011	1110100

Flowchart:

Dies Flussdiagrammen stellt den Ablauf des Codes prinzipiell dar.



Die Entschlüsselung wird hierbei von For-Loops übernommen, dies ist mit abstand nicht die Effizienteste Methode dies zu Decodieren, jedoch wurde sich für diese Variante entschieden da, während dem Decodieren nichts „Kritisches“ passiert und es grundlegend egal ist wie lange dieser Vorgang dauert. Das Decodieren selbst wurde in zwei Bereiche aufgeteilt, einmal den Bereich vor dem Mittleren Guard-Bits und den danach. Da abhängig davon auf zwei verschiedene Lookup-Tables gegriffen werden muss.

> *Eine vielfach bessere Methode, zur Decodierung wäre es, die Sieben erkannten Bits der einzelnen Zahlen in einen String oder Char Array (...) zu speichern und den „Lookup-Table“ auch mit String/Chars zu gestallten. Dadurch lassen sich die ineinander verpackten For-Loops vermeiden. Und das Entschlüsseln schneller ausführen da nicht jedes einzelne Bit verglichen wird, sondern nur geschaut wird, ob den der Inhalt jeder String (7-Bit) dem Inhalt mit dem verglichenen String entspricht.*
> *Selbes lässt sich mit Int erreichen dabei könnte man zum Dekodieren dann auch einen switch verwenden.*
>
> *Im Code auf GitHub finden sich drei verschieden Ansätze zur Lösung der Dekodierung, zum Zeitpunkt der Abgabe, ist jedoch nur eine zu einem Theoretisch funktionierenden Punkt ausprogrammiert.*

Github:

https://github.com/05TapPau/ArduinoBarcodeReader.git

Quellen:

https://en.wikipedia.org/wiki/EAN-8

https://www.dcode.fr/barcode-ean8

https://softmatic.com/de/barcode-ean-8.html