

DIY: Garage X-Ray

Valerie Wiedemann
 Universität Erlangen-Nürnberg
 Email: valerie.wiedemann@fau.de
<https://github.com/vwiedemann/DIY-Garage-X-Ray>

Zusammenfassung—Bei nicht fest zugewiesenen Garstellplätzen ist die Wahl des Parkplatzes dann problematisch, wenn das Garagentor geschlossen ist und der ankommende Fahrer nicht weiß, hinter welchem Tor sich ein freier Parkplatz verbirgt.

Das Projekt „Garage X-Ray“ bietet dem Parkplatzsuchenden eine Lösung, indem es die Parkplatzbelegung sichtbar macht, ohne die Garagentore öffnen zu müssen.

Die Umsetzung erfolgte durch 2 Module, die jeweils über Mikrocontroller gesteuert werden und per Funk miteinander kommunizieren.

I. MOTIVATION

Jede Familie, die über weniger Garagenparkplätze als Autos verfügt, kennt das Problem: Man möchte in der Garage parken, öffnet das Tor, doch dort steht allerdings schon ein Auto. Man muss also warten, bis sich das gerade geöffnete Tor wieder geschlossen hat und sich danach einen anderen Parkplatz suchen. Sind mehrere Garagenparkplätze vorhanden, verstärkt sich dieses Problem, da der Vorgang gegebenenfalls öfter durchgeführt werden muss.

Ziel des Projekts Garage X-Ray war es, einen Mechanismus zu entwickeln, der es ermöglicht, herauszufinden, ob in der Garage ein Auto steht, ohne das Garagentor dafür öffnen zu müssen.

II. PROJEKTBESCHREIBUNG

Prinzipiell gibt es mehrere Ansätze, dieses Problem zu lösen. So könnte man außen an der Garage eine Art Ampel anbringen, die den Garagenstatus anzeigt. Diese Variante bringt aber mit sich, dass an der Vorderfront der Garage eine Ampel angebracht werden muss. Dies beeinflusst das Erscheinungsbild und wurde somit für dieses Projekt ausgeschlossen. Außerdem kann bei der Ampel-Lösung jeder vorbeilaufende Spaziergänger den Status der Garage sehen. Dadurch kann jeder Fremde darauf schließen, ob jemand zuhause ist. Das wiederum ist sicherheitstechnisch nicht optimal.

Deswegen wurde für die Umsetzung des Projekts eine Lösung mit 2 Modulen favorisiert, wie sie in Abbildung 1 zu sehen ist. Ein kleines Modul wird im Auto

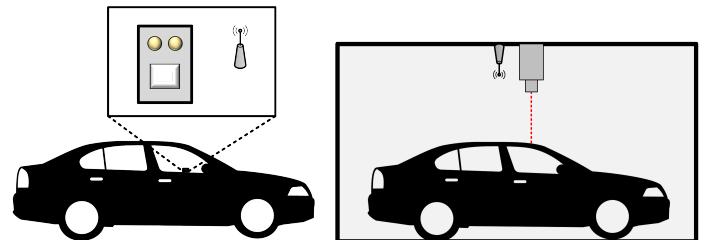


Abbildung 1: Projektaufbau

platziert und das andere in der Garage. Per Funk kann vom Auto-Modul aus dann die Belegung beim Garagen-Modul abgefragt werden. Je nach Ergebnis leuchtet die LED des Auto-Moduls entweder Rot oder Grün. Ob sich ein Auto in der Garage befindet, ermittelt das Garagen-Modul über einen Sensor, welcher die Entfernung misst. Auch das Garagen-Modul ist mit einem Funk-Modul versehen, damit es sowohl feststellen kann, ob gerade eine Anfrage vom Auto-Modul gestellt wird, als auch die Antwort zurückübermitteln kann.

Die Anforderungen an das Projekt waren dabei, das Auto-Modul möglichst klein und kompakt zu halten, damit man es problemlos in der Mittelkonsole aufbewahren kann. Außerdem soll die angedachte Lösung für das Garagenmodul aus möglichst wenigen Teilen bestehen, damit dieses leicht anzubringen ist und möglichst wenige Kabel durch die Garage gezogen werden müssen.

III. MATERIALLISTEN

Um das Projekt zu realisieren, werden folgende Materialien benötigt, sofern ein Modul für genau einen Parkplatz gebaut werden soll. Die beiden Module können natürlich so erweitert werden, dass es möglich ist, auch noch einen zweiten oder dritten Parkplatz damit zu überwachen. Hier wird der Übersichtlichkeit halber davon ausgegangen, dass es sich nur um die Überwachung eines einzigen Stellplatzes handelt.

A. Auto-Modul

- 1 Mikrocontroller ATMEGA88-PA

- 1 Sockel für den Mikrocontroller
- 1 Quarz 16 MHz
- 2 Kondensatoren 18 pF
- 2 Batterien CR 2032
- 1 Tastschalter
- 1 Rot-Grün LED
- 1 PNP Transistor BC307C
- 1 NPN Transistor BC548C
- Widerstände:
 - 1 1 kΩ
 - 1 10 kΩ
 - 2 220 kΩ
 - 1 33 kΩ
- 1 Spannungsregler LM2931 AZ
- 1 Keramik-Kondensator 0.1 µF
- 1 Elko 100 µF
- 1 Transceivermodul nRF24L01+
- Zum Testen: 1 Arduino Uno

B. Garagen-Modul

- 1 Arduino (in diesem Projekt wurde ein Arduino Mega verwendet, aber ein kleinerer Arduino reicht auch aus)
- 1 Ultraschallsensor HC-SR04
- 1 Transceivermodul nRF24L01+

IV. AUTO-MODUL

Das Auto-Modul besteht aus mehreren Elementen, auf die im Folgenden genauer eingegangen wird. Zentraler Bestandteil ist der Mikrocontroller, welcher die Aktivitäten des Moduls steuert. Neben diversen Bauteilen, die dazu dienen, den Mikrocontroller mit Strom zu versorgen, enthält das Auto-Modul noch ein Funk-Modul für die Kommunikation und eine LED, deren Farbe dem Nutzer dann den Status der Belegung mitteilt.

A. Selbsthaltung

Da das Wechseln von Batterien zu den Dingen gehört, die lästig sind und die man eher ungern durchführt, war es Ziel des Projekts, den Stromverbrauch des Auto-Moduls möglichst gering zu halten. Optimalerweise sollte die Schaltung nur dann Strom verbrauchen, wenn auch tatsächlich eine Abfrage stattfindet. Dies lässt sich natürlich sehr einfach mit einem Schalter realisieren. Da es aber nicht sonderlich benutzerfreundlich ist, das Modul an- und ausschalten zu müssen, wurde eine Selbsthaltungsschaltung installiert. Diese sorgt dafür, dass der Mikrocontroller mit Strom versorgt wird, sobald der Taster gedrückt wird und die Stromversorgung so lange anhält, bis der Mikrocontroller sich selbst abschaltet.

Die erste Idee hierfür war, die Selbsthaltung über eine Darlington-Schaltung umzusetzen. Diese wird durch Hintereinanderschalten von zwei PNP-Transistoren realisiert. Beim Ausprobieren auf dem Steckbrett stellte sich jedoch heraus, dass eine Darlington-Schaltung für diesen Zweck nicht geeignet ist, da zu viel Spannung an den Transistoren abfällt und es mit dieser Variante nicht möglich ist, aus den 6 V, die die Batterien liefern, die 5 V zu bekommen, die der Mikrocontroller braucht. Mosfet-Transistoren konnten auch nicht eingesetzt werden, da hier das gleiche Problem bestand und die Diode der Mosfets außerdem hinderlich wäre. Nachdem die Darlington-Schaltung aus den genannten Gründen nicht funktionierte, wurde sie durch eine Schaltung mit einem PNP- und einem NPN-Transistor ersetzt, welche auch komplementäre Darlington-Schaltung genannt wird.

Die Bipolartransistoren sind, wie in Abbildung 2 zu sehen, miteinander verdrahtet. Bei Q1 handelt es sich um den PNP-Transistor BC307C, Q2 ist ein NPN-Transistor BC548C. Der Emitter von Q2 ist an die Basis von Q1 angeschlossen und steuert somit Q1.

Wird der Tastschalter S1 betätigt, fließt Versorgungsspannung zum Mikrocontroller und er startet. Die erste Aktion des Mikrocontrollers ist es, einen festgelegten Ausgangspin auf High und damit 5 V zu setzen, der über einen Widerstand an der Basis von Q2 angeschlossen ist. Dies führt dazu, dass Q2 schaltet und die GND-Spannung an der Basis von Q1 angelegt wird, wodurch auch Q1 schaltet. Dadurch fließt nun die Versorgungsspannung über Q1 zum Mikrocontroller, der Schalter ist überbrückt und kann losgelassen werden. Wenn der Mikrocontroller mit dem restlichen Programm fertig ist, setzt er den Steuerungspin wieder auf Low, wodurch die Stromversorgung unterbrochen wird. Somit fließt kein Strom mehr, bis der Taster erneut betätigt wird.

B. Spannungsregler

Um das Auto-Modul möglichst klein zu halten, werden zwei Knopfzellenbatterien mit jeweils 3 V als Stromquelle genutzt. Da der Mikrocontroller eine Eingangsspannung von 5 V erwartet, wird diese mit Hilfe eines Spannungsreglers auf 5 V geregelt.

Vor den Spannungsregler ist ein 0.1 µF Kondensator geschaltet, der in Abbildung 3 als C1 zu sehen ist. Dieser fungiert als Tiefpassfilter und sorgt dafür, dass ungewollte Spannungsspitzen nicht beim Spannungsregler ankommen. Hinter den Spannungsregler ist ein zweiter Kondensator C2 geschaltet, der die Aufgabe übernimmt, dass keine unbeabsichtigten Schwingungen am Ausgang des Spannungsreglers entstehen.

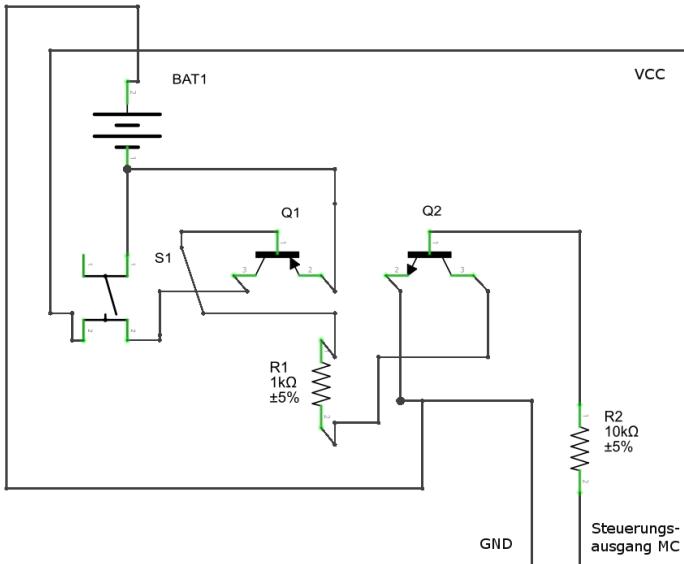


Abbildung 2: Selbsthaltungsschaltung

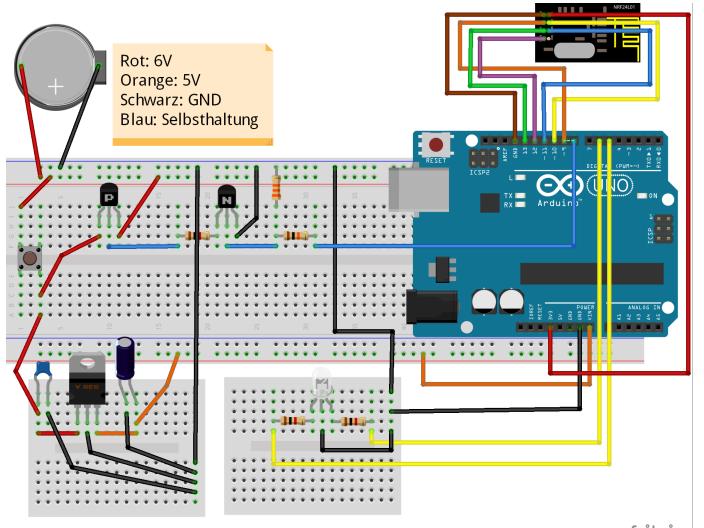


Abbildung 4: Testaufbau für Selbsthaltungsschaltung und Spannungsregler

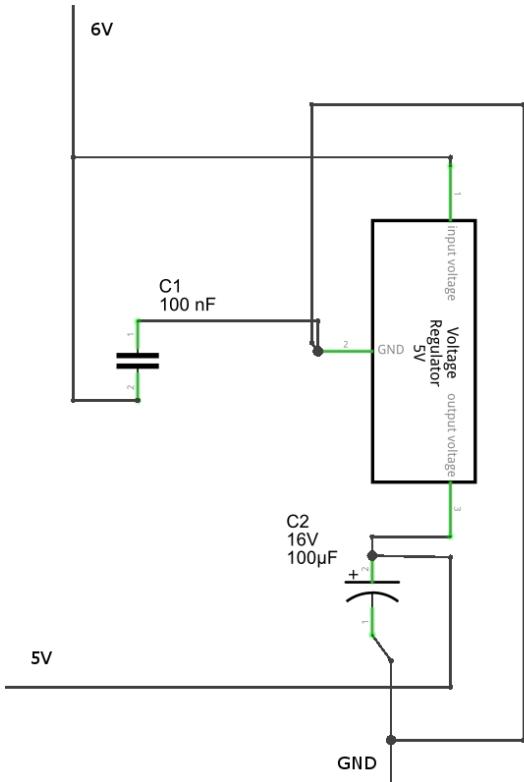


Abbildung 3: Spannungsregler

C. Testaufbau

Um die Funktionalität des Auto-Moduls und das Zusammenspiel der einzelnen Komponenten zu testen, wurde die in Abbildung 4 dargestellte Schaltung aufgebaut. Hierbei empfiehlt es sich dringend, die Anschlüsse erst einmal mit dem Multimeter durchzumessen, bevor man sie tatsächlich an den Arduino anschließt. Schnell einmal ist ein Bauteil falsch angeschlossen und dies

kann zu starken Spannungen führen, welche den Arduino beschädigen können.

Abgesehen von der Selbsthaltungsschaltung und dem Spannungsregler, besteht das Auto-Modul noch aus einer Rot-Grün-LED und einem Funkmodul, welches zur Kommunikation mit dem Garagenmodul dient. Auf die Informationsübertragung der Komponenten wird im Abschnitt „Kommunikation“ genauer eingegangen.

Ein Durchlauf des Auto-Moduls sieht folgendermaßen aus: Durch Drücken des Schalters wird der Stromkreis geschlossen, über den Spannungswandler fließt Strom zum Arduino. Dieser setzt *Pin8* auf *HIGH*, wodurch die Selbsthaltung aktiviert wird. Anschließend sendet der Arduino über das Funkmodul eine Anfrage an das Garagenmodul. Dieses schickt die Information zurück, ob sich gerade ein Auto in der Garage befindet oder nicht. Je nachdem steuert der Arduino *Pin5* oder *Pin6* an, wodurch die LED entweder rot, für belegt, oder grün, für frei, leuchtet. Nach einigen Sekunden wird die LED ausgeschaltet und der Mikrocontroller kappt seine Stromversorgung, indem er den Steuerpin für die Selbsthaltung, *Pin8*, auf *LOW* setzt.

V. GARAGEN-MODUL

Das Garagen-Modul setzt sich aus einem Arduino, einem Funkmodul und einem Ultraschallsensor zusammen. Da in der Garage eine konstante Stromversorgung über die Steckdose problemlos möglich ist, kann der Arduino dauerhaft angeschlossen bleiben und es muss nicht darauf geachtet werden, den Stromverbrauch zu minimieren, wie beim Auto-Modul. In Abbildung 5 ist der Aufbau des Garagen-Moduls dargestellt, auf den in

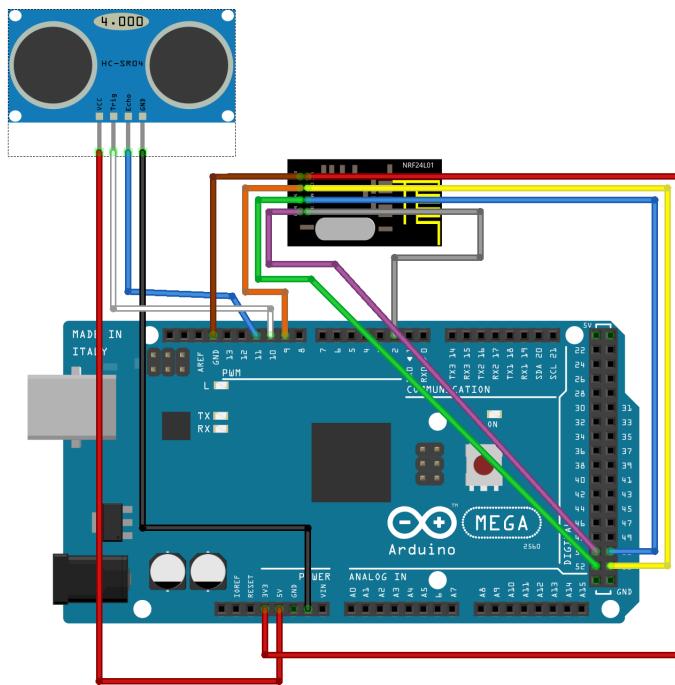


Abbildung 5: Aufbau des Garagen-Moduls

den folgenden Abschnitten nochmals genauer eingegangen wird.

A. Ultraschallsensor

Um zu bestimmen, ob sich in der Garage ein Auto befindet, kamen verschiedene Methoden in Frage. Denkbar wäre hier eine Kamera, die das Nummernschild erkennt, wodurch auch noch eine Identifizierung des jeweiligen Autos möglich wäre. Diese Lösung ist aber für den einfachen Zweck, herauszufinden, ob der Parkplatz belegt ist oder nicht, etwas überdimensioniert. Naheliegend wäre auch eine Lichtschranke. Bei dieser Variante muss allerdings auf der gegenüberliegenden Seite des Sensors ein Reflektor oder ein Empfänger befestigt werden, was in diesem Projekt vermieden werden sollte, um mit möglichst wenigen zusätzlichen Kabeln und Elementen in der Garage auszukommen. In Parkhäusern werden oft Induktionsspulen im Boden verbaut, um die Präsenz eines Autos festzustellen. Da der Garagenboden allerdings schon fertig gefliest ist, ist es nicht möglich, die Spule im Boden zu befestigen. Sie einfach auf den Boden zu legen birgt ein Verletzungspotential.

Deshalb fiel die Wahl auf das günstig zu erhaltende Ultraschall-Modul HC-SR04. Dieses kann laut Datenblatt [1] Distanzen zwischen 2-400 cm messen.

Ausgelöst wird die Messung über eine fallende Flanke am Trigger-Anschluss, wie Abbildung 6 zeigt. Daraufhin sendet der Sensor einen 40 kHz Ultraschallpuls aus und empfängt dessen Echo. Das Ergebnis der Messung wird

Ultraschallmodul	Arduino
VCC	5 V
GND	GND
TRIG	11
ECHO	10

Tabelle I. Pinbelegung des Ultraschallmoduls

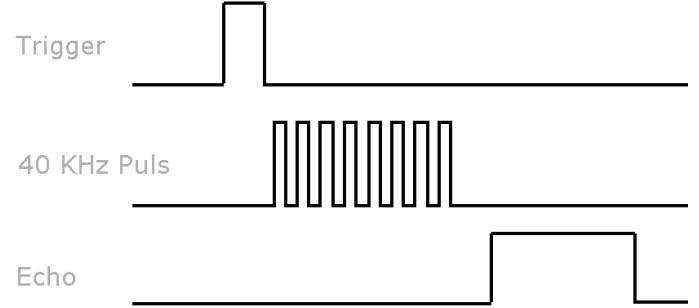


Abbildung 6: Ablauf eines Messvorgangs des Ultraschallsensors

über den Echo-Pin mitgeteilt, welcher so lange auf High gesetzt wird, wie die Messung gedauert hat. Diese Dauer wird über den Arduino gemessen und daraus anschließend anhand folgender Formel der Abstand berechnet:

$$Distance = \frac{Duration}{29 \frac{\mu s}{cm}} \cdot \frac{1}{2} \quad (1)$$

Schall legt bei einer Raumtemperatur von 15 °C ungefähr eine Distanz von $340 \frac{m}{s}$ zurück, was $29 \frac{\mu s}{cm}$ entspricht. Da der Schall am Gegenstand, auf den er trifft, zurückgeworfen wird und somit die Zeit doppelt gemessen wird, muss das Ergebnis halbiert werden. In der Garage herschen über das Jahr große Temperaturschwankungen. Schall ändert bei einer anderen Umgebungstemperatur seine Geschwindigkeit, sodass das Messergebnis dann etwas von der Realität abweicht. Dies kann allerdings für dieses Projekt vernachlässigt werden, da die temperaturbedingten Abweichungen 5% nicht übersteigen und für die Abstandsmessung eine Genauigkeit von 10 cm ausreicht.

Der Mikrocontroller löst einmal pro Minute eine Messung aus, berechnet den Abstand des nächsten Hindernisses zum Ultraschallsensor und schließt daraus, ob ein Auto vorhanden ist oder nicht. Da die Garagenhöhe 250 cm beträgt und ein Auto ungefähr 150 cm hoch ist, wird davon ausgegangen, dass sich bei einer Entfernung von weniger als 120 cm ein Auto in der Garage befindet, bei größeren Werten nicht. Den Belegungsstatus speichert der Mikrocontroller bis zur nächsten Messung, bei der er diesen wieder überschreibt.

Der Grund dafür, warum das Garagen-Modul die Festlegung des Status übernimmt und den Auto-Modulen nicht einfach die Entfernung übermittelt, ist, dass so nur der Arduino in der Garage angepasst werden muss, falls sich an der Methode zur Entfernungsbestimmung etwas ändern sollte. Denn ein Arduino ist schnell umprogrammiert, beim Auto-Modul müsste erst der Mikrocontroller ausgebaut werden. Außerdem sind von den Auto-Modulen mehrere vorhanden, Garagen-Modul gibt es aber nur eines.

Die Abstandsmessung wird nur einmal in der Minute durchgeführt, ansonsten befindet sich der Controller im Schlafmodus. Eine Minute ist zwar ein relativ langer Zeitraum. Da ein Auto aber eine gewisse Zeit braucht, um aus der Garage herauszufahren, wird es von dem gerade ankommenden Autofahrer sofort gesehen. Somit kann davon ausgegangen werden, dass es zu keinen Fehlern kommt, auch wenn nur einmal pro Minute gemessen wird.

B. Gehäuse

Da eine Garage beileibe keine klinisch reine Zone ist, ist es wichtig, die Module zu schützen. Für diesen Zweck wurde ein Gehäuse für den Ultraschallsensor erstellt. Nach einigen eigenen Design-Versuchen fiel die Wahl hierbei auf das optisch schön gestaltete Gehäuse von Thingiverse [4]. Dieses wurde mit dem 3D Drucker im Fablab mit einer Druckzeit von ca 4 Stunden erstellt und passt perfekt auf den Sensor. Für den Druck wurden sowohl Gehäuse als auch Deckel so gedruckt, wie sie im rechten Teil von Abbildung 7 liegen. Dies geschah im Gegensatz zur Vorlage, die einen Druck um 90° gedreht vorschlägt. Die gewählte Druckrichtung erwies sich als praktikabel. Für den Gehäuseteil wurden Stützstrukturen unter den beiden Nasen am Rand gedruckt. Dadurch sind diese nicht ganz eben, was aber später nicht mehr auffällt. Denn diese Stellen werden später von den Schrauben überdeckt.

Auch für die Knopfbatterien wurde ein Gehäuse gedruckt, um die Batterien ohne Wackelkontakt mit dem Testaufbau verbinden zu können. Hier findet sich ebenfalls ein formschönes Gehäuse auf Thingiverse [3], welches Platz für zwei CR 2032 Batterien bietet. In den hinteren Teil des Gehäuses wird nach dem Drucken eine Buchsenleiste mit 4 Anschlüssen gesteckt. Diese wird, wie in der linken Bildhälfte von Abbildung 8 zu sehen, mit zwei Drahtstücken verlötet, die den Kontakt zwischen den Anschlüssen und der Batterie herstellen.

VI. KOMMUNIKATION

Wie in den vorangehenden Abschnitten schon erwähnt, erfolgt der Datenaustausch zwischen dem Auto-

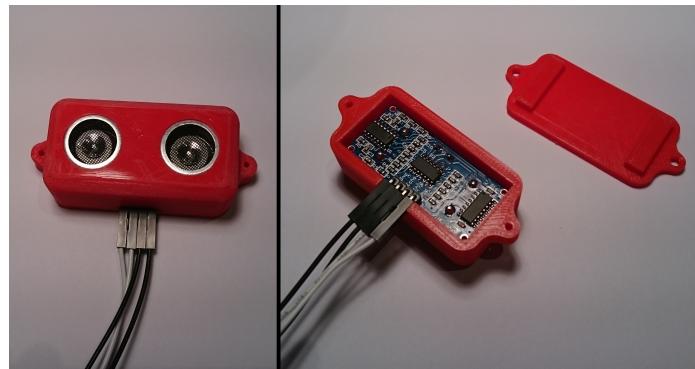


Abbildung 7: gedrucktes Ultraschallsensor-Gehäuse



Abbildung 8: gedrucktes Batterie-Gehäuse

Modul und dem Garagen-Modul über das Funk-Modul nRF24L01+ [8]. Die Wahl fiel auf dieses Transceiver-Modul, da somit mit nur einem Bauteil sowohl das Senden als auch das Empfangen möglich sind und dies eine kompaktere Bauweise erlaubt, als es mit einem Transmitter- und einem Receiver-Modul möglich gewesen wäre. Beim nRF24L01+ handelt es sich um Transceiver-Modul vom Hersteller Nordic Semiconductors, welches sowohl senden als auch empfangen kann. Der nRF24L01+ sendet mit einer Frequenz von 2.4 GHz und ist für einen niedrigen Energieverbrauch ausgelegt.

Die Reichweite des Moduls wird mit bis zu 100 m angegeben, was natürlich nur unter optimalen Bedingungen möglich ist. Ein Test in der Garage ergab ein zuverlässiges Signal bis zu einer Reichweite von 10 m. Möchte man die Reichweite vergrößern, kann der Chip noch um eine externe Antenne erweitert werden, für den geplanten Einsatz sind 10 m jedoch völlig ausreichend.

Die Einrichtung der Kommunikation orientierte sich an dem Tutorial der Arduino-Info-Website[2].

Als erstes müssen die Funk-Module an den Mikrocontrollern angeschlossen werden. Die jeweiligen Pin-Belegungen sind Tabelle II zu entnehmen. Zu beachten ist hierbei, dass das Funk-Modul mit einer Versorgungsspannung von 3.3 V arbeitet und nicht wie der Arduino mit 5 V. Die anderen Eingänge des Moduls sind

5 V tolerant, sodass man sie ohne Zwischenschaltung mit dem Arduino verbinden kann. Wie man an den Namen der Pins schon erkennen kann, kommunizieren die nRF24L01+-Module über das Serial Peripheral Interface, kurz SPI. Dieses Bus-System regelt die synchrone Kommunikation zwischen einem Master und beliebig vielen Slaves. Für die Kommunikation über SPI werden 4 Leitungen benötigt:

Das Clock-Signal (SCK), der Master Output Slave Input (MOSI), der Master Input Slave Output (MISO), das Chip Select Not Signal (CSN). Das Clock-Signal wird vom Master erzeugt und an die Slaves übertragen, sodass alle einen einheitlichen Takt haben. Indem er das jeweilige Chip Select Not Signal auf GND zieht, kann der Master auswählen, mit welchem der Slaves er kommunizieren möchte. Die Kommunikation über SPI selbst zu implementieren ist möglich. Jedoch ist dies relativ fehleranfällig, wenn man die Pin-Ansteuerung zur Datenübertragung mittels der GPIO-Primitiven der Arduino-Bibliothek implementieren muss. Für die Arduino IDE [7] existiert die Bibliothek RF24 [6], welche das Kommunikationsprotokoll implementiert und es dem Anwender erlaubt, seine Anweisungen auf einer höheren Ebene zu formulieren. Die Bibliothek ist noch relativ neu, wird aber aktiv gepflegt und kann in der Arduino IDE über „Add Library“ eingebunden werden. Die Bibliothek bietet viele Einstellungsmöglichkeiten, so kann beispielsweise die Datenrate angepasst werden oder die Größe der zu übermittelnden Pakete. Eine erste Kommunikation ist jedoch auch mit den Standard-Einstellungen möglich und so kann man den Code nach und nach den eigenen Bedürfnissen anpassen.

Im Garage X-Ray Projekt übernimmt das Auto-Modul die Rolle des Masters. Nach einem kurzen Setup sendet das Auto-Modul einen Request an das Garagen-Modul. Dieser Request besteht aus einer fest definierten Zahl, welche dafür sorgen soll, dass das Garagen-Modul auch nur auf Anfragen antwortet, die tatsächlich vom Auto-Modul kommen. Der nRF24L01+ des Garagen-Moduls empfängt die Nachricht und löst über seinen Interrupt-Pin IRQ einen Interrupt beim Arduino aus. Dieser Interrupt sorgt dafür, dass der Arduino aus seiner Mess- und Schlaf-Routine herauptspringt und den ankommenden Interrupt bearbeitet. Wenn es sich bei dem Interrupt um eine Anfrage des Auto-Moduls handelt, antwortet das Modul mit einem Acknowledgement. In der Bibliothek kann festgelegt werden, dass Acknowledgements nicht einfach nur leere Nachrichten sind, sondern auch Daten enthalten. So wird also mit dem Acknowledgement gleich der Belegungs-Status mitübertragen. Diese Variante hat den Vorteil, dass die beiden Kommunikationspartner in ihrem jeweiligen Modus bleiben können und nicht zwischen

„Lesend“ und „Schreibend“ wechseln müssen.

nRF24L01	Arduino Mega	Arduino Uno
GND	GND	GND
VCC	3.3 V	3.3 V
CE	9	9
CSN	53	10
SCK	52	13
MOSI	51	11
MISO	50	12
IRQ	2	2

Tabelle II. Pinbelegung des Ultraschallmoduls

VII. AKTUELLER STAND UND AUSBLICK

Das Auto-Modul ist funktionsfähig. Es existiert zur Zeit aber nur auf einem Steckbrett und muss dementsprechend noch auf eine Platine aufgelötet werden. Die anfängliche Überlegung, SMD-Bauteile einzusetzen wird nicht umgesetzt, da die aktuell verwendeten Bauteile in „normaler“ Größe nicht viel Platz brauchen und das Modul klein genug ist.

Das Garagen-Modul ist komplett funktionsfähig und bereit für die Montage. Da das Modul 2 Parkplätze überwachen soll, wird es in der Mitte der Garage befestigt und jeweils durch ein altes, vierdriges Telefonkabel mit den Ultraschallsensoren verbunden.

Durch die Verwendung des Transceiver-Funkmoduls ist eine Erweiterung des Projekts dahingehend denkbar, dass über das Auto-Modul auch gleich der Garagentor-Öffner angesteuert wird. Diese Funktion wird aber zur Zeit von den Stakeholdern nicht gewünscht, da die Wahl des Parkplatzes einem komplexen „Wer-muss-wann-wieder-weg“-Zeitplan folgt, der zu kompliziert und nichtdeterministisch ist, um ihn mit einer Maschine zu steuern.

LITERATUR

- [1] Datenblatt HCSR04, www.micropik.com/PDF/HCSR04.pdf, 01.02.2016.
- [2] Tutorial NRF20L01, http://www.mikrocontroller.net/articles/NRF24L01_Tutorial, 01.02.2016.
- [3] Batteriegehäuse, <http://www.thingiverse.com/thing:25625>, 01.02.2016.
- [4] Gehäuse Ultraschallsensor, <http://www.thingiverse.com/thing:1170335>, 01.02.2016.
- [5] Kommunikation über SPI, <http://www.mct.de/faq/spi.html>, 02.02.2016.
- [6] Arduino-Bibliothek für den nRF24L01+, <http://tmrh20.github.io/RF24/>, 02.02.2016.
- [7] Arduino IEE, <https://www.arduino.cc/en/Main/Software>, 02.02.2016.
- [8] nRF24L01+ Datenblatt, www.nordicsemi.com/eng/content/download/2726/34069/file/nRF24L01P_Product_Specification_1_0.pdf, 02.02.2016.
- [9] komplementäre Darlington-Schaltung, <http://www.elektronik-kompendium.de/public/schaerer/kdarl1.htm>, 02.02.2016.