Ein Bild, das Tisch, Kalender enthält.

Automatisch generierte BeschreibungEinheit 2: Bluetooth Verbindung

Materialien

* 2x Arduino Nano (+Verbindungskabel)
* 2x Bluetooth Modul HC-05
* 2x Breadboards (mittelgroß)
* 2x Breadboards (klein)
* Jumperkabel männlich/männlich und weiblich/männlich

Abbildung Materialien

Im Alltag kommt man ständig mit Bluetooth in Verbindung, aber wie ist das eigentlich technisch umgesetzt? Die grundlegende Kommunikation wird über Bluetooth-Module umgesetzt, Kleine in sich funktionierende Chips, die das Suchen, Verbinden und die Übertragen zu großen Teilen übernehmen und automatisieren.

Es gibt zwei verschiedene Modul Typen, die verschiedene Stellungen übernehmen, denn bei Bluetooth Verbindungen gibt es immer einen Master, der die Verbindung einleitet und die Kommunikation übernimmt und einen Slave, der nur „ausgenutzt“ wird und die Verbindung und Daten annehmen muss. Das HC-06 ist ein reines Slave-Modul, dass keine eigenen Verbindungen aufbauen kann, sondern nur bspw. von einem Handy Daten empfangen kann. Das HC-05 Modul hingegen kann sowohl als Master- als auch Slave-Modul genutzt werden und ist deshalb perfekt für eine Kommunikation in beide Richtungen.

**Ein Bild, das Text, Schaltung enthält.

Automatisch generierte Beschreibung**Das HC-05 Modul besitzt sechs verschiedene Pins: Der GND und VCC-Pin sind für die Stromversorgung zuständig, dabei wird der GND-Pin mit dem GND-Pin des Arduinos verbunden und der VCC-Pin mit dem 3V Pin des Arduino. Der State Pin kann an einen beliebigen Digital Pin angeschlossen werden und gibt ein High/Low-Signal, je nachdem ob aktuell eine aktive Verbindung existiert oder nicht. Der RXD- und TXD-Pin sind die Datenpins des Moduls und werden vertauscht mit den RX0- und TX1- Pin des Arduinos verbunden. Der EN-Pin ist ein Konfigurations-Pin und standardmäßig nicht verbunden.

Bluetooth Modul anschließen

****Der Arduino wird standardmäßig auf dem mittelgroßen Breadboard angebracht. Das HC-05 Modul wird auf dem kleinen Breadboard angebracht, damit das passt, muss man den EN-Pin etwas nach oben biegen.

Unbedingt darauf achten, dass das HC-05 Modul mit dem 3V Pin des Arduinos verbunden wird. Auf keinen Fall an den 5V Pin anschließen. Das kann zu schweren Schäden an dem HC-05 Modul führen.

**Aufgabe 1**

|  |  |
| --- | --- |
| **1** | **Befestige den Arduino und das HC-05 Modul auf ihren jeweiligen Breadboards und verbinde die Pins korrekt.** |

Konfiguration

Das Ziel ist es zwei Arduinos über Bluetooth miteinander kommunizieren zu lassen. Dazu müssen die Bluetooth-Module so konfiguriert werden, dass sie sich finden und automatisch koppeln können. Dabei sollen die Module auch nicht von anderen Geräten gefunden werden können. Um ein HC-05 Modul konfigurieren zu können müssen wir das Modul in den Konfigurationsmodus bringen, dazu verbinden wir den EN-Pin mit dem Arduino an Digital-Pin 3, dann können wir mit dem Skript später den Arduino in den AT-Modus bringen. Dieser Modus lässt sich erkennen daran, dass die LED des HC-05 Moduls anders blinkt als zuvor.

Die Kommunikation des HC-05 Moduls funktioniert über die gleichen Kanäle wie die serielle Kommunikation, daher kann es zu gegenseitigen Störungen kommen. Um das zu verhindern kann man einen virtuellen RX/TX Kanal erstellen. Dazu nutzen wir die Bibliothek „SoftwareSerial“ und können dann mit dem einen neuen seriellen Kanal als Variable definieren, dabei sind dann die beiden angegebenen Pins, die neuen virtuellen RX und TX-Pins. Verbinde also den RX-Pin des HC-05 Moduls jetzt mit Digital-Pin 5 des Arduinos und den TX-Pin des HC-05 Moduls mit dem Digital-Pin 4 des Arduinos. Eine beispielhafte Anwendung siehst du im Skript, dass wir für den Konfigurationsmodus brauchen.

Wenn das Modul sich im AT-Modus befindet, solltest du im seriellen Monitor den Text „Enter AT commands:“ lesen können und wenn du „AT“ eingibst und sendest, sollte ein „OK“ zurückkommen, dann hat alles korrekt funktioniert. Wenn es zu einem Fehler gekommen ist, versuch es noch einmal oder hole dir Hilfe bei deinem Lehrer.

Es gibt nun ein paar Einstellungen, die im AT-Modus eingestellt werden müssen. Zuerst interessiert uns die Adresse des Moduls, diese kriegen wir mit dem Befehl „AT+ADDR?“, als Antwort sollte dann die Adresse des Moduls erscheinen. Notier dir diese. Danach wollen wir die Rolle des Moduls einstellen. Wir haben zwei Module, eines davon soll als Master eingestellt werden, das andere als Slave. Mit dem Befehl „AT+ROLE=“ kann man die Rolle einstellen, eine 0 an das Ende gehangen ändert die Rolle des Moduls zu einem Slave-Modul, eine 1 an das Ende gehangen ändert die Rolle des Moduls zu einem Master-Modul. Mit dem Befehl „AT+CMODE=0“ können wir dem Modul befehlen, sich mit niemanden zu verbinden, außer der verbundenen Adresse, die wir im nächsten Schritt angeben. Im besten Fall weißt du jetzt die Adressen von beiden Modulen, denn im nächsten Schritt müssen wir beiden Modulen jeweils die Adresse des anderen übergeben. Mit dem Befehl „AT+BIND=“ befehlen wir dem Modul, mit welcher Adresse es sich verbinden soll. Häng einfach die Adresse des anderen Moduls an das Ende des Befehls.

#include <SoftwareSerial.h>//Bibliothek für virtuellen Kanal

SoftwareSerial BTSerial(4, 5); //name(RX-Pin, TX-Pin)

void setup()

{

  pinMode(3, OUTPUT);

  digitalWrite(3, HIGH);

  Serial.begin(38400);

  Serial.println("Enter AT commands:");

  BTSerial.begin(38400);

}

void loop()

{

  if (BTSerial.available()) Serial.write(BTSerial.read());

  if (Serial.available()) BTSerial.write(Serial.read());

}



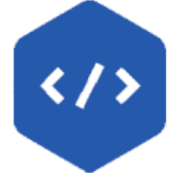
Wenn es beim Hochladen zu Problemen kommt, kann das an dem HC-05 Modul liegt, dass verursacht manchmal Probleme beim Hochladen. Einfach den RX/TX Pin für die Dauer des Hochladen herausziehen, dann sollte alles wieder funktionieren.

**Aufgabe 2**

|  |  |
| --- | --- |
| **2a** | **Verbinde den EN-Pin und den RX/TX-Pin des HC-05 Moduls an die korrekten Pins des Arduinos.** |
| **2b** | **Lade den oben angegebenen Code auf den Arduino hoch.** |
| **2c** | **Konfiguriere beide deiner HC-05 Module**   * **AT+ADDR?** * **AT+ROLE=*0/1*** * **AT+CMODE=0** * **AT+BIND=*ADDR*** |

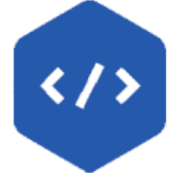
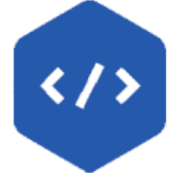
Verbindung und Kommunikation

Wenn beide HC-05 Module korrekt konfiguriert wurden sollten diese sich nun automatisch verbinden, wenn beide eingeschaltet und kopplungsbereit sind.

Wie bereits im Text zu Aufgabe 2 beschrieben, nutzen wir bei der Verwendung von HC-05 Modulen häufig virtuelle serielle Kommunikation. Diese kann effektiv die gleichen Befehle durchführen wie die echte serielle Verbindung (BTSerial.print()/BTSerial.println()). Bei der Kommunikation von rohen Daten über die Bluetooth Verbindung nutzt man eher den Befehl BTSerial.write(). Um die serielle Kommunikation zu nutzen muss sie wieder gestartet werden, diesmal beide Serial.begin() und BTSerial.begin() beide mit der Geschwindigkeit 38400.

|  |  |
| --- | --- |
| **BTSerial.write(array, array\_length);** | |
| **Beschreibung** | Dieser Befehl wird genutzt, um Rohdaten, die nicht von dem Arduino verändert werden sollen, über die Bluetooth-Verbindung zu verschicken. |
| **Parameter** | array: Informationen, die über die Bluetooth-Verbindung geschickt werden sollen, gespeichert in einem Array. Es können nur positive Zahlen geschickt werden, wollen also negative Zahlen übertragen werden, müssen die vorher in positive Zahlen umgewandelt werden und nach der Übertragung wieder zurück.  array\_length: Die Anzahl an Elementen, die in dem Array enthalten sind |
| **Rückgabewert** | Gibt die Anzahl an Zeichen zurück, die tatsächlich geschrieben wurden, falls es zu Problemen mit dem Schreiben kommen sollte. |

Der Slave, der die Daten bekommt, will in der loop() nun wissen, ob Daten über die Bluetooth Verbindung angekommen sind und nutzt dafür den BTSerial.available() Befehl.

Für das eigentliche Auslesen der Rohdaten geschieht dann über den BTSerial.readBytes() Befehl.

|  |  |
| --- | --- |
| **BTSerial.readBytes(array, array\_length);** | |
| **Beschreibung** | Dieser Befehl wird genutzt, um angekommene Rohdaten aus dem Bluetooth-Modul auszulesen. |
| **Parameter** | array: Das Array, in das die Daten reingeschrieben werden sollen  array\_length: Die Anzahl der Elemente die gelesen werden sollen |
| **Rückgabewert** | Gibt die Anzahl an Bytes zurück, die tatsächlich gelesen wurden. |

|  |  |
| --- | --- |
| **BTSerial.available();** | |
| **Beschreibung** | Dieser Befehl wird genutzt, um abzufragen, ob Bluetooth Daten verfügbar sind zu lesen. |
| **Parameter** | void |
| **Rückgabewert** | Gibt einen bool zurück, true, wenn Daten verfügbar sind, false, wenn nicht |

**Aufgabe 3**

|  |  |
| --- | --- |
| **3a** | **Verbinde beide Arduinos mit ihrem jeweiligen HC-05 Modul.** |
| **3b** | **Code schreiben**   * **Master Sketch**   + **Erstelle einen virtuellen seriellen Kanal**   + **Schicke einmal ein char Array mit den Werten 1, 2 und 3**   + **Schicke einmal ein char Array mit den Werten -1, 0 und 1**   + **Gib jeweils alle geschickten Werte auch auf dem seriellen Monitor aus** * **Slave Sketch**   + **Erstelle einen virtuellen seriellen Kanal**   + **Teste, ob Daten über die Bluetooth-Verbindung angekommen sind**   + **Wenn Daten vorhanden sind, lies sie in ein vorher angelegtes Array ein**   + **Gib die angekommenen Daten auf dem seriellen Monitor aus** |

Aktuellen Status auslesen

Der letzte Pin, den wir von dem HC-05 Modul bisher noch nicht verwendet haben ist der State-Pin. Dieser Pin kann bei Bluetooth-Verbindungen, die auf eine aktive Verbindung angewiesen sind, ganz hilfreich sein, um zu erkennen, ob die Verbindung noch besteht. Wenn an einen Digital-Pin angeschlossen gibt er dauerthaft HIGH/LOW-Werte, je nach Verbindungsstatus.

**Aufgabe 4**

|  |  |
| --- | --- |
| **4a** | **Verbinde den State-Pin mit einem Digital-Pin des Arduinos.** |
| **4b** | **Code schreiben**   * **Beide Arduinos sollen auf dem seriellen Monitor „True“ ausgeben, solange eine aktive Bluetooth-Verbindung besteht** * **Es soll „False“ ausgegeben werden, wenn die Verbindung nicht mehr besteht** |

Das Modul braucht immer zwischen 3-5 Sekunden bis der State-Pin aktualisiert wird, also bei kleinen Verzögerungen nicht verwirrt sein.