Mobiles Hardware-Praktikum 2023

Marc Pfeifer, Dr. Philipp Scholl, Professur für Rechnerarchitektur,
Institut für Informatik, Albert-Ludwigs-Universität Freiburg, Freiburg im Breisgau

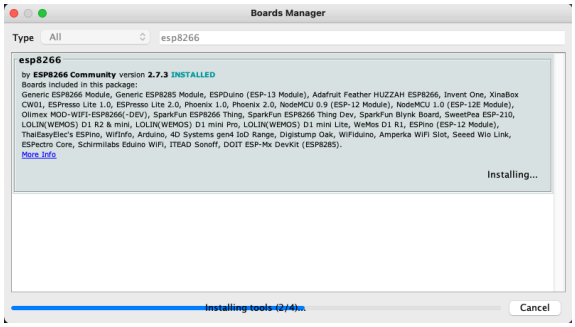
Übungsblatt 2 (26 Punkte)

Ziel:

Mit Hilfe des ESP8266 werden die Motoren und Ultraschall-Sensoren des S-Trike angesteuert und auf einer Website angezeigt. Der ESP8266 wird auch mit der Arduino IDE programmiert. Dies ist allerdings eine andere Komponente als der in der vorhergehenden Aufgabe verwendete Arduino.

Vorbereitung:

ESP8266: Damit der ESP8266 über die Arduino IDE programmiert werden kann, müssen Sie zunächst einige Voreinstellungen vornehmen. Öffnen Sie die Einstellungen von Arduino und fügen Sie in das Feld "Additional Boards Manager URLs" die folgende URL hinzu: `https://arduino.esp8266.com/stable/package_esp8266com_index.json`. (Haben Sie dort bereits eine URL eingetragen, müssen Sie jede weitere mit einem Komma separieren.
Öffnen Sie nun den Board Manager unter *Tools* → *Board* → *Boards Manager* und geben Sie in die Suche "esp8266" ein. Installieren Sie die neueste Version des esp8266 Arduino cores wie in folgender Abbildung gezeigt.



Wählen Sie unter *Tools* → *Board* → *ESP8266 Boards* "LOLIN (WEMOS) D1 R2 & Mini" und unter *Tools* → *Port* den korrekten COM-Port aus. Lassen Sie außerdem, wenn nicht anders beschrieben, die weiteren Board Einstellungen unter *Tools* auf Standard. Gibt es nachfolgend beim Upload des Programms Probleme, sollte der *Upload Speed* auf maximal "921600 baud" eingestellt werden.
Um sicherzustellen, dass Sie den ESP8266 nun programmieren können, verbinden Sie den ESP über das Micro USB Kabel mit ihrem PC. Für den CH340 USB-Chipsatz sollten keine weiteren Treiber benötigt werden. Öffnen Sie nun *File* → *Examples* → *01.Basics* → *Blink* und laden Sie das Besipiel auf den ESP. Die LED auf der Unterseite des ESP8266 sollte nun mit einer Frequenz von 1Hz blinken.

Motoren: Die Motoren des Roboters werden über eine H-Brücke angesteuert. Diese erlaubt es die Drehrichtung der Motoren frei zu wählen. Zur Steuerung der H-Brücke werden zwei Signale pro Motor benötigt. Diese sind an die Buchse "Motors" auf der Experimentierplatine geleitet. Die genaue Ansteuerung der Motoren können Sie der Dokumentation der Experimentierplatine (in der S-Trike Dokumentation) in ILIAS entnehmen.

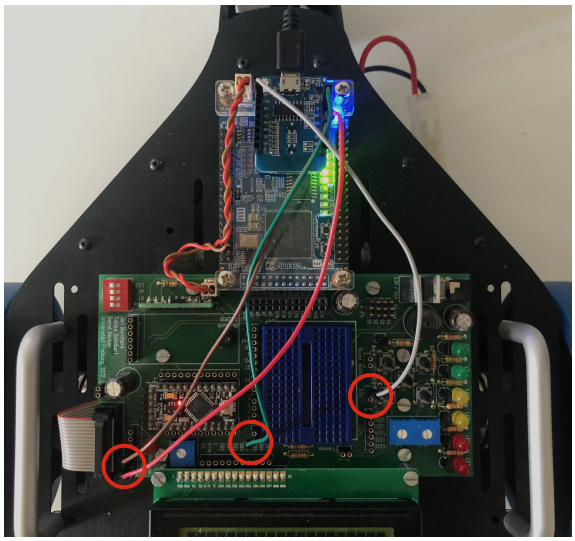
Ultraschallsensoren: Die Ultraschallsensoren werden jeweils über eine Leitung angesprochen. Auf dieser Leitung muss zunächst der *trigger* vom Microcontroller an den Sensor gesendet werden um

eine Messung zu starten. Der Sensor antwortet darauf auf der gleichen Leitung nach einer Verzögerung mit einem Puls. Die Länge des Pulses korreliert mit der Entfernung des eines Objekts zum Sensor.

Hinweis: Bitbreiten von Variablen können je nach Microcontroller-Architektur unterschiedlich breit sein. Ein “int” hat z.B. auf dem Arduino Pro Mini nur 16 Bit, und auf dem ESP8266 32 Bit. Bitte nutzen Sie Typenbezeichner mit Größenangabe (also z.B. int8_t, int16_t und int32_t bzw. uint8_t, . . .). “float” und “double” haben die übliche Größe (32 bzw. 64 Bit) sollten aber nur genutzt werden wenn sie wirklich benötigt werden, da Berechnungen mit Fließkommazahlen aufwendiger sind. Dies wird auch bei der Korrektur der Übungsblätter entsprechend bewertet.

Aufgabe 1 (3 Punkte):

Verbinden Sie außerdem die Buchse “GND” auf dem Experimentierboard mit der Buchse “G” sowie die Buchse 3V3 mit 3.3V. Diese Verbindung sollte immer bestehen, wenn Sie den ESP benutzen. Achtet auch darauf das der Roboter eingeschaltet ist und die Räder keinen Kontakt mit dem Boden haben. Erstellen Sie ein neues Arduino Programm für den ESP8266. Definieren Sie die Pins D1 und D2 als Ausgang und verbinden Sie sie mit den Motor-Signalen A1 und A2.



Fügen Sie eine neue Funktion “setMotor” zu Ihrem Programm hinzu. Diese soll als Parameter einen Wahrheitswert (bool) “forward” erhalten. Implementieren Sie dann die folgende Funktionalität in der Funktion: Hat “forward” den Wert $true$, soll der Motor sich vorwärts drehen, sonst rückwärts. Erzeugen Sie dazu die entsprechenden Steuersignale für den Motor.

Hinweis:

- Da alle Roboter unterschiedlich verkabelt wurden, muss die korrekte Belegung von A1 und A2 zum vorwärts und rückwärts Drehen selbst herausgefunden werden.
- Damit der Motor sich dreht, muss jeweils eines der beiden Steuersignale $HIGH$ (1) und das andere LOW (0) sein.
- Wenn das richtige Board ausgewählt ist (LOLIN (WEMOS) D1 mini), kann man die Macros D0-D8 verwenden, um Pin Operationen durchzuführen (Beispiel: *pinMode(D1, OUTPUT)* setzt D1 als Ausgang). Zusätzlich kann man dem Pinout des D1 Mini dem entsprechenden Dokument in ILIAS entnehmen (D1 ist hier bspw. GPIO 5).

Aufgabe 2 (2 Punkte)

Um die Geschwindigkeit des Motors zu regulieren müssen Sie das $HIGH$ Signal schnell zwischen 1 und 0 umschalten. Die Länge der 1-Phase bestimmt dann die Geschwindigkeit des Motors. Dieses Verfahren ist als Pulsweitenmodulation (PWM) bekannt.

Auf dem Arduino sowie auf dem ESP steht ihnen dazu die Funktion “analogWrite(pin, value)” zur Verfügung. Während diese beim Arduino Pro Mini einen 8-bit Wert erwartet, ist die Bitbreite 10-bit für den ESP. Dementsprechend erzeugt die Funktion für den ESP ein PWM-Signal zwischen *konstant aus* ($value = 0$) und *konstant an* ($value = 1023$).
Erweitern Sie die Funktion “setMotor” um einen weiteren Parameter “speed” (uint16_t) und implementieren Sie die Funktion so, dass die Rotationsgeschwindigkeit vom Signal “speed” abhängt. Ihre Funktion sollte den Motor mit gegebener Geschwindigkeit in beide Richtungen rotieren lassen können.

Aufgabe 3 (2 Punkte):

Verbinden Sie Motor B mit Pins D5 und D6 des ESPs und erweitern Sie “setMotor” um einen weiteren Parameter, um auszuwählen, welcher Motor angesprochen werden soll.

Fügen Sie eine neue Funktion “drive” zu Ihrem Programm hinzu. Diese erhält drei Parameter. Ein flag, um auszuwählen ob vorwärts oder rückwärts gefahren werden soll, eine Zeit in Millisekunden und eine Geschwindigkeit. Die Funktion soll den Roboter für die gegebene Zeit geradeaus (vorwärts oder rückwärts) fahren lassen und dann stoppen.
Sie können den “delay(time)” Befehl verwenden um die entsprechende Verzögerung zu erreichen. Nutzen Sie die gerade implementierte “setMotor” Funktion.

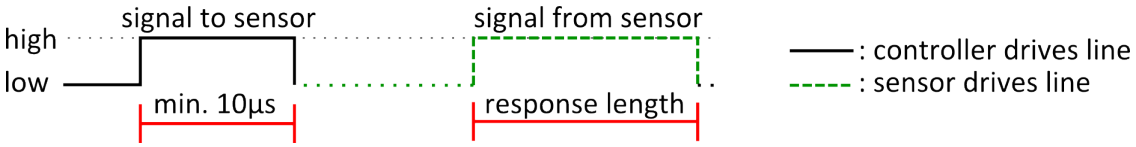
Aufgabe 4 (2 Punkte):

Fügen Sie eine neue Funktion “turn” zu Ihrem Programm hinzu. Diese erhält diesselben Parameter wie die “drive” Funktion. Das flag soll hierbei auswählen, ob der Roboter sich rechts oder linksherum drehen soll. Die Drehung soll “auf der Stelle” erfolgen.
Benutzen Sie wieder die “setMotor” Funktion.

Aufgabe 5 (5 Punkte):

Verbinden Sie den Ultraschall-Sensor US1 mit Pin D8 des ESP8266. Fügen Sie eine neue Funktion “measureDistance” zu ihrem Programm hinzu. Diese soll die gemessene Distanz zu einem Hindernis in Metern zurückgeben (oder -1 wenn kein Hindernis erkannt wurde). Übergen Sie den Ultraschall-Sensor-Pin als Parameter.
Um den Sensor zu aktivieren muss zunächst der Sensor-Pin als Ausgang definiert werden und ein Trigger-Impuls gesendet werden (*LOW*, dann mindestens 10 μs *HIGH*, dann wieder *LOW*). Danach muss der Pin als Eingang definiert werden. Der Sensor hält jetzt die Leitung für eine bestimmte Zeit auf *LOW* und sendet dann ebenfalls einen *HIGH*-Puls. Messen Sie die Pulsweite.

Hinweis: Sie können die “pulseIn”-Funktion benutzen.

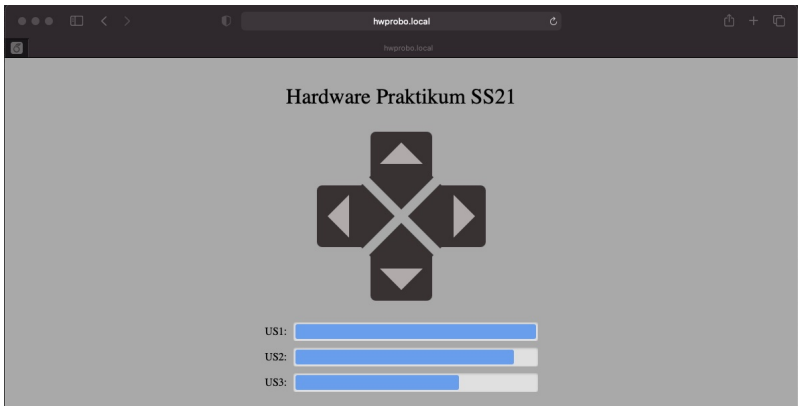


Wenn Sie nach 30 ms keinen Puls vom Sender erhalten haben, wurde kein Objekt detektiert. Die Entfernung in cm hat einen linearen Zusammenhang mit der Pulsweite. Finden Sie diesen und schreiben sie dazu einen kurzen Kommentar in ihren Code.
Hinweis: Nutzen Sie zum Testen der Funktion die serielle Konsole.

Aufgabe 6 (6 Punkte)

Öffnen Sie den mitgegebenen Sketch “RoboWebServer.ino”.
Der Sketch erstellt einen HTTP Server auf Port 80. Verbindet sich ein Client auf diesem Port (zB. über einen Browseraufruf), wird ein entsprechender HTML, CSS und Javascript Code gesendet, welcher eine interaktive Webseite anzeigt.

Damit die Website auch wirklich interaktiv wird, müssen Sie den Code aber noch entsprechend anpassen.



Ändern Sie zunächst die Konstanten “ssid” und “password”, sodass sich der ESP mit Ihrem WiFi Netzwerk verbindet. Laden Sie dann das Programm auf den ESP.
Hinweise:

- Eduroam und andere Enterprise WiFis werden nicht unterstützt.
- Wenn Sie gerade kein entsprechendes WiFi Netzwerk verfügbar haben, können Sie zB. mit jedem modernen Smartphone selbst ein Netzwerk erstellen.
- Sollte der ESP sich trotz mehrfachem Prüfen der Zugangsdaten nicht mit dem WiFi Netzwerk verbinden lassen, resetten Sie den Controller und laden den Sketch erneut hoch.

Unter seiner IP Adresse (siehe serielle Konsole) oder dem mDNS Namen “hwprobo.local” sollte die Webseite nun erreichbar sein. Testen Sie dies mit dem Browser Ihrer Wahl bevor sie fortfahren (getestet unter Chrome und Safari).

Fügen Sie nun alle bisher implementierten Funktionen (*drive*, *turn* und *measureDistance*) zu diesem Sketch hinzu. Verbinden Sie außerdem die beiden anderen Ultraschallsensoren, wie im Sketch beschrieben (US2 auf D7 und US3 auf D3).

Die Webseite kommuniziert über *GET* Requests mit dem ESP. Diese GET Requests müssen auf dem ESP8266 verarbeitet werden und die entsprechende Funktionalität implementiert werden.
Die Webseite pollt automatisch jede 500ms nach neuen Ultaschall Werten. Im Arduino Code ist bereits an der entsprechenden Stelle eine if Abfrage vorgesehen, um den GET Request für das Polling abzufangen. Dort müssen Sie die Ultraschallsensor-Werte messen und in die Variablen *us1*, *us2* und *us3* speichern.
Für die Implementierung des Steuerkreuz (D-Pad), müssen äquivalent die entsprechenden GET Requests abgefangen werden und die entsprechende Funktionalität implementiert werden. Werden die einzelnen Richtungen des Steuerkreuzes gedrück, soll der Roboter vorwärts oder rückwärts fahren, oder sich auf der Stelle nach links oder rechts drehen. Pro Druck sollte er das natürlich nur für eine kurze Zeit zB 300ms machen. Indem Sie die Tasten des Steuerkreuz drücken und die Ausgabe in der seriellen Konsole beobachten, können sie herausfinden, wie die entsprechenden GET Requests aussehen.

Aufgabe 7 (6 Punkte)

Der HTML, CSS und Javascript Code ist in der Datei “website.h”. Diese Datei sollte auch als zweites Tab in der Arduino IDE angezeigt werden.
Fügen Sie einen weiteren Button “Tesla Mode” auf der Webseite hinzu. Um den entsprechenden GET Request zu senden, muss das HTML Element eine *id* erhalten und die *onClick* Methode sollte auf “gamepad(this)” gesetzt werden (wie bei den entsprechenden Buttons des D-Pads).
Der GET Request des Buttons muss nun wieder auf dem ESP8266 abgefangen werden. Es soll dabei die folgende Funktion implementiert werden. Wird der Button gedrückt, soll der Roboter von selbst losfahren. Erkennt der Roboter ein Hinderniss, soll er diesem selbständig ausweichen. Wird der Button erneut gedrückt, soll der Roboter wieder stoppen.

Abgabe:

Archivieren Sie Ihr Programm in einer ZIP-Datei und laden Sie diese hoch. Ihre Abgabe sollte ein Programm mit einer Lösung für jede Aufgabe enthalten. Überprüfen Sie die Archiv-Datei auf Vollständigkeit.