

## GPS Tracker

### Ziel

Einen GPS (Global Positioning System) Tracker schreiben und einen Track im KML (Keyhole Markup Language) Format auf der integrierten  $\mu$ SD Karte aufzeichnen. Die Übung ist zweiteilig: Im ersten Teil erstellen Sie die Software im Labor, für den zweiten Teil nehmen Sie den GPS Tracker im Akkubetrieb mit und zeichnen einen Track auf. Vor der nächsten Übung schicken Sie mir den Quelltext des Programms, sowie den aufgezeichneten KML-Track per Email. Benötet wird diesmal also erst später.

### Vorgehensweise

Schließen Sie das GPS Shield per USB an den Rechner an. Der Akku ist nicht angeschlossen. SD Karte und Knopfzelle sind eingelegt.

Machen Sie sich nun mit der Hardware vertraut, indem Sie das Tutorial lesen. Das GPS Shield enthält ein GPS Modul, welches seriell angesteuert wird und sogenannte NMEA (National Marine Electronics Association) Kommandos an den Arduino zurückgibt. Aus den NMEA Zeichenketten lassen sich unter anderem Uhrzeit und Position auslesen. Probieren Sie die Funktionsfähigkeit des Shields im „Direct Connect“ Modus – per serielltem Monitor (9600 Baud) sollten Sie die Rohdaten der GPS Engine zu sehen bekommen. Achtung: Einen „Fix“ (die Feststellung der Position) bekommen Sie nur am Fenster, eventuell nicht innerhalb des Raums.

Probieren Sie dann das „Parsing“ Beispiel aus dem Tutorial aus. Dazu müssen Sie die **GPS** Bibliothek installieren. Das GPS Modul wird nun per Software angesteuert (damit bleibt der serielle Hardware Port für USB frei – der Arduino hat nur einen UART). Damit dies funktioniert, vergessen Sie nicht, die Pinbelegung mittels `mySerial(8, 7)` zu ändern.

Falls Sie Ihr Programm hübsch machen wollen, ersetzen Sie den Code für den Timer durch die **MsTimer2** Bibliothek. Dieses Beispiel zeigt Ihnen, wie man auf dem Arduino die NMEA Codes auswerten kann und ist die Basis für das Schreiben der KML Dateien. Achten Sie auf die hohe Verbindungsgeschwindigkeit (115200 Baud) bei der Benutzung des Beispiels.

Probieren Sie als nächstes den Tutorial-Code zur Ansteuerung der SD Karte. Achtung: Sie müssen keine SD Bibliothek herunterladen, die korrekte Version sollte bereits vorhanden sein. Achten Sie auch auf die Änderung der Zeile mit der **SD.begin(...)** Funktion. Wenn alles funktioniert, sollte das Programm die NMEA Codes direkt auf die SD Karte schreiben. Sie können die SD Karte auslesen und die Daten z.B. über die „GPS Visualizer“ Website anschauen.

Sie kennen nun die Funktionalität des Shields. Schreiben Sie jetzt ein Programm, welches beim Drücken auf den Taster die rote LED anschaltet und anfängt die Daten im KML Format auf die  $\mu$ SD Karte zu schreiben. Bei einem zweiten Tasterdruck stoppt die Aufzeichnung, die LED erlischt und die KML Datei wird abgeschlossen. Der Taster befindet sich an Pin 4 und die LED an Pin5. Beide sind mit inverser Logik geschaltet. Eine Schablone für die KML Datei finden Sie in Moodle. Die Schablone kann komplett übernommen werden, es müssen nur die Koordinaten in Zeile 18 in der Form **Längengrad,Breitengrad,Höhe** ersetzt werden. Als Trenner für die Nachkommastellen wird der Punkt verwendet, der Trenner zwischen zwei Koordinaten ist das Leerzeichen. Achten Sie darauf die Angaben in Grad zu schreiben. Achten Sie ebenfalls

darauf, dass die Daten mit größtmöglicher Präzision geschrieben werden – verlieren Sie „unterwegs“ keine Nachkommastellen! Um KML Dateien auf die SD Karte zu schreiben, müssen Sie die Schablone im Speicher halten. Verwenden Sie dazu das **PROGMEM** Schlüsselwort, bzw. das **F ( )** Makro, so landen die Daten im Flash und nicht im RAM.

Zum Abschluss testen Sie Ihren Tracker. Lösen Sie die USB Verbindung und schließen Sie den Akku an (er sollte mindestens zwei Stunden halten). Zeichnen Sie einen interessanten Track auf und kontrollieren Sie das Ergebnis in Google Earth.

### Vorbereitung

Beschäftigen Sie sich mit Grundlagen und Einschränkungen des GPS. Verstehen Sie das NMEA und KML Format, sowie die zur Verfügung gestellte KML Vorlage. Lesen Sie die Tutorials, sowie die Dokumentation der verwendeten Bibliotheken und Funktionen. Machen Sie sich mit den empfohlenen Websites vertraut.

### Achtung

Denken Sie an den Pullup-Widerstand für den Taster. Unter Linux müssen die Bibliotheken bei der Installation umbenannt werden (keine Bindestriche). Achten Sie bei den Beispielen auf Ihre Arduino Version – es gibt Unterschiede zwischen Uno und Leonardo.

### Wichtige Funktionen & Bibliotheken

- PROGMEM und das **F ( )** Makro
  - <http://playground.arduino.cc/Learning/Memory>
- Die Adafruit GPS Bibliothek
- Die SD Bibliothek
- Die MsTimer2 Bibliothek

### Sie brauchen

- Standalone Arduino Board, USB Kabel, 9V Blockakku, Adapter zum Anschluss des Akkus, GPS-Shield mit Knopfzelle, µSD Karte mit Adapter, µSD Lesegerät
- Tutorial: <https://learn.adafruit.com/adafruit-ultimate-gps-logger-shield>
- Google Earth: <https://www.google.de/intl/de/earth/>
- Adafruit GPS Library: <https://github.com/adafruit/Adafruit-GPS-Library>
- MsTimer2 Library: [http://www.pjrc.com/teensy/td\\_libs\\_MsTimer2.html](http://www.pjrc.com/teensy/td_libs_MsTimer2.html)
- GPS Visualizer: <http://www.gpsvisualizer.com>

### Notengebung

4,0 (Anwesend); 3,0 (GPS Ausgabe wird geparsed); 2,3 (Es wird ein Track im KML Format geschrieben); 2,0 (+ die Daten sind plausibel und bis auf die Genauigkeit fehlerfrei); 1,7 (+Code sauber geschrieben und dokumentiert); 1,3 (+ Flash für Strings verwendet); 1,0 (+Bestmögliche Genauigkeit)