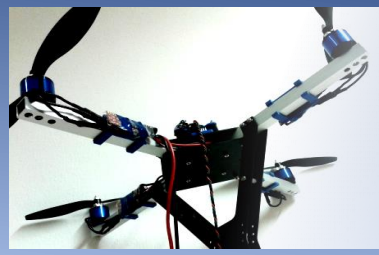
Matlab

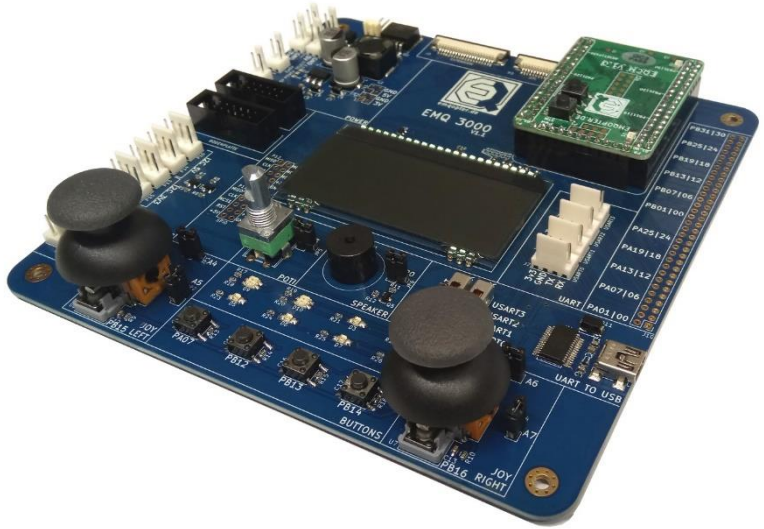
Telemetrie & Telekommand

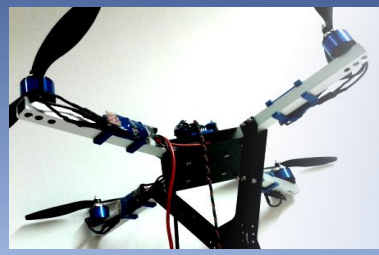


Inhalt

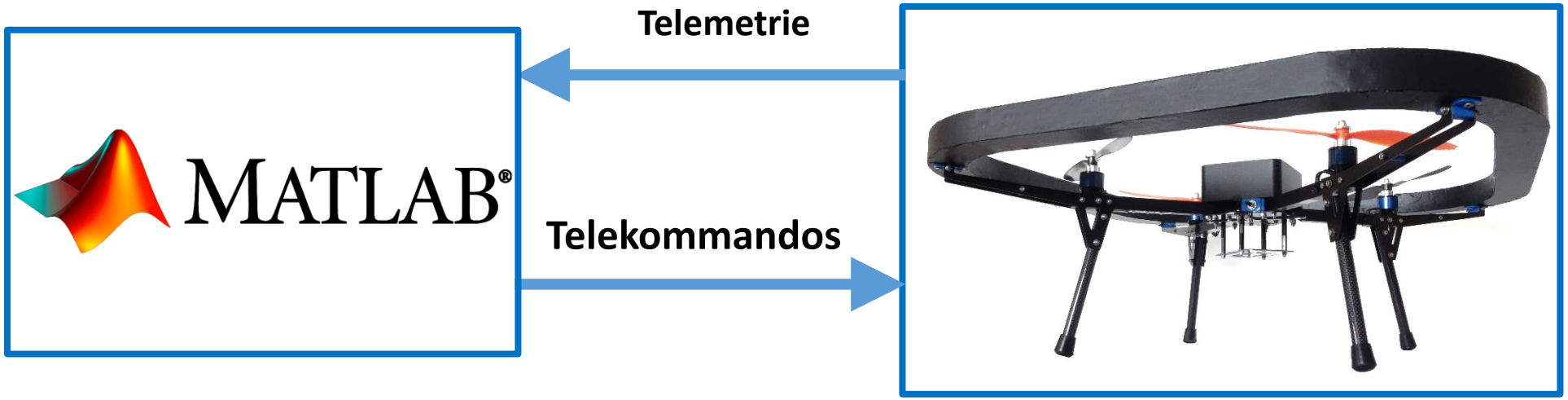
Umfang: ca. 2-3 Zeitstunden

- Matlab Telemetrie & Telekommand
- Matlab: Serielle Kommunikation
- Matlab: Daten plotten
- Matlab: Eigene Funktionen erstellen
- Aufgaben



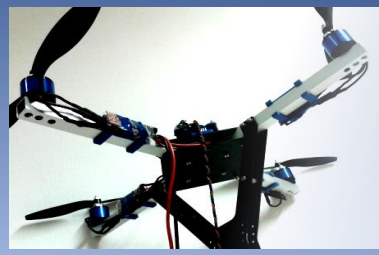


Matlab Telemetrie & Telekommand



Pilot / Bodenstation
Auswertung / Optimierung

Quadrokopter

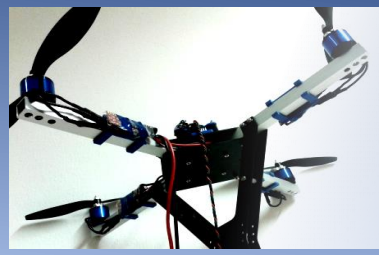


Matlab Telemetrie & Telekommand

- Die bereits bekannte Telemetrie wurde erweitert, um mit Matlab kommunizieren zu können.

- Im Ordner COM des Projekts „EMQ_QCSF_Matlab“ befinden sich die Dateien:
  - matlab_protocol.h: legt das Kommunikationsprotokoll fest
  - matlab_telemetry.c/.h: sendet regelmäßig aktivierte Daten an Matlab
  - matlab_telecommand.c/.h: liest und führt Telekommandos von Matlab aus

- Von der Funktionsweise und deren Aufbau sind sie der bereits bekannten Telemetrie ähnlich. Lediglich die Syntax, also die Struktur der Kommunikation wurde angepasst.

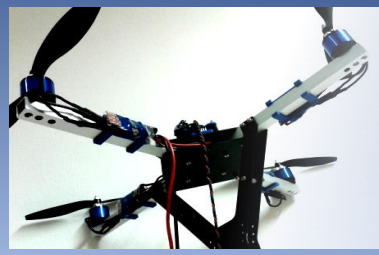


Matlab: Serielle Kommunikation

- In Matlab gibt es seit der Version R2019b die Klasse Serialport, mit der serielle Schnittstellen in Matlab abgebildet und angesteuert werden können.

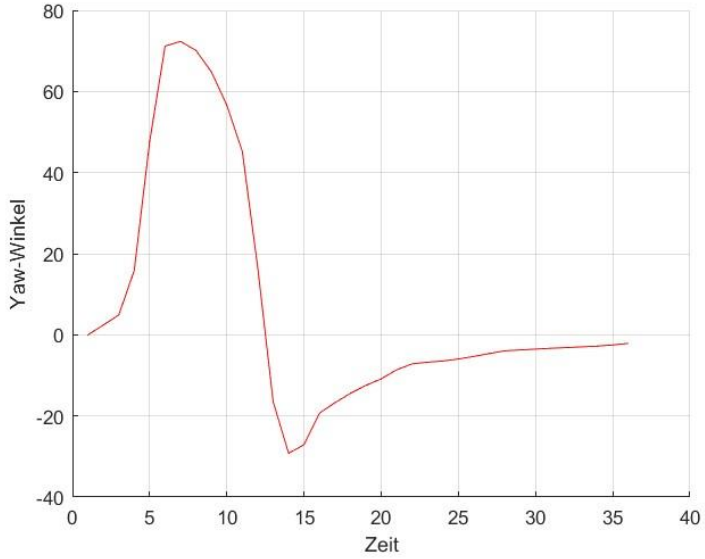
- Erzeugt man ein Objekt der Klasse Serialport, kann über den entsprechenden COM-Port des PCs mit dem QCS/F mittels USART kommuniziert werden.

- Benötigte Funktionen:
  - Init: device = serialport(port, baudrate, Timeout, Value);
  - Daten senden: writeline(device, data);
  - Daten empfangen: data = readline(device);
  - Buffer leeren: flush(device,“input/output“);
  - Verbindung trennen: clear device;

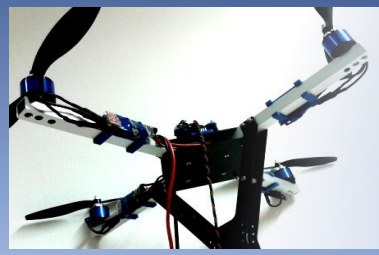


Matlab: Daten plotten

Benötigte Funktionen:



- Figur erzeugen: figur;
- Gitter aktivieren: grid on;
- Neue Daten in gleiche Figur eintragen: hold on;
- Achsenbeschriftung: ylabel(„Y-Beschriftung“);
  xlabel(„X-Beschriftung“);
- Daten plotten: plot(data, farbe, etc);

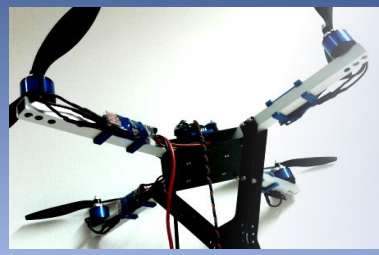


Matlab: Eigene Funktionen erstellen

- Neue Matlab-Datei im Workspace anlegen, die den gleichen Namen wie die Funktion hat (bsp. myfun.m)

```
function [y1, y2] = myfun(x1,x2)
        y1 = x1;
        y2 = x2;
    end
```

- Am Ende der Funktion werden die Rückgabewerte (y1, y2) automatisch an das aufrufende Programm übergeben.



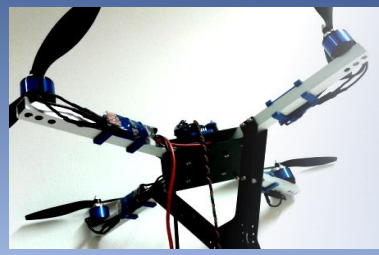
Aufgaben

Notwendige Hardware:

- EMQ3000
- QCS / QCSF
- Mini USB Kabel für USART-Verbindung und zum Flashen

Notwendige Software:

- Matlab Version >= R2019b, keine zusätzlichen Toolboxen
- AVR Studio 32 installiert (mit Tool Chain und FLIP Treiber)
- Projekt-Code und Library:
  EMQ_QCSF_Matlab



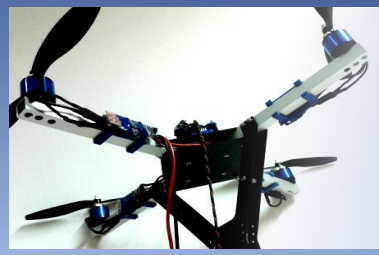
Aufgaben

Hilfe und Hintergrund:
Die Aufgaben sind mit Hilfe des Projekts EMQ_QCSF_Matlab zu lösen. Legen Sie sich zusätzlich einen eigenen Dateipfad für Ihre Matlab-Dateien an und fügen Sie diesen Ihrem Matlab Workspace hinzu.

Aufgabe 1:
Legen Sie eine Datei main.m an und initialisieren Sie die Schnittstelle zum EMQ 3000 Board mit Hilfe der Klasse Serialport. Beachten Sie dabei, dass die Baudrate 57600 beträgt und der Timeout mit 0.5 gesetzt wird. Am Ende des Programmes soll der Serialport wieder freigegeben werden.
*) Warum ist es sinnvoll einen Timeout zu wählen?

Aufgabe 2:
Machen Sie sich mit dem Aufbau der Matlab-Telemetrie und dem Matlab-Telekommand vertraut. Welcher Syntax folgt die jeweilige Struktur der Kommunikation?

Hinweis: Schauen Sie sich die Datei matlab_protocol.h im Ordner Com auf AVR-Seite an.



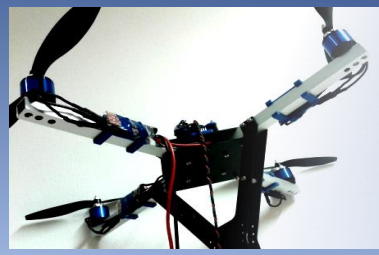
Aufgaben

Aufgabe 3:

Implementieren Sie eine MATLAB-Funktion init_Telemetry.m, mit der die Telemetrie für die PID-Parameter und die RPY-Daten aktiviert und deaktiviert werden kann. Aktivieren Sie dann in main.m die Telemetrie für die RPY-Daten.

Hinweis: Schauen Sie sich die Methoden writeline und flush der Klasse Serialport an und finden Sie heraus, welche Werte die Methode matlab_transcommand erwartet. Um Fehler in der Kommunikation zu vermindern, sollte die Methode zweimal hintereinander mit einer Pause dazwischen ausgeführt werden.

*) Warum werden durch diese Maßnahmen Fehler minimiert?



Aufgaben

Aufgabe 4:

Nachdem nun die Telemetrie für die RPY-Daten aktiviert wurde, soll ein Echtzeitplot des Yaw-Winkels implementiert werden. Nutzen Sie zum Auslesen des Yaw-Winkels entweder die vorgegebene Funktion read_Telemetry.m oder schreiben Sie Ihre eigene (für Fortgeschrittene). Nun können Sie den Yaw-Winkel des QCS per Hand verstellen und sehen den Winkel als Funktion der Zeit in Matlab.

Hinweis bei eigener Implementierung:
Schauen Sie sich die Methode readline an, um Daten vom Usart auszulesen. Um Die Telemetry-Daten zu extrahieren, kann die Methode split verwendet werden. Machen Sie sich mit globalen Variablen in Matlab vertraut und beachten Sie die Hinweise in den vorgegebenen Funktionen.