Fachhochschule Nordwestschweiz

Dokumentation

Datenübertragung vom SBGC zum SBC

Serielle Kommunikation von Phytec-Recher und Simple Brushless Gimbal Controller

17. Oktober 2016

Saner Kevin

Institut für Automation

1 Einleitung

Die Antennen zum Suchen der Minen sollen mit einer Lagenregelung versehen werden, dazu wird auf ein Gimbal-System gesetzt. Die Konfiguration der Lagenregelung soll dabei vor dem Flug erfolgen. Das heisst, dass während dem Flug keine Änderungen vorgenommen werden müssen, jedoch soll die Winkel der Gimbal-Regelung überwacht werden, damit man sie allenfalls in das Postprocessing miteinbeziehen kann. Die im folgenden beschriebene Software, lässt also folglich nur Lese- und keine Schreibprozesse zu.

Voraussetzungen:

- Ubuntu Version 14.04 oder ähnlich
- SimpleBG 32-bit Extended controller
- USB-Kabel (Typ-A - Mini-B)
- Toolchain des Single Board Computer zum Cross-Kompilieren
- Level shifter Low-To-High (bidirektional)
- optional: 1 x FTDI-Kabel 5V
- optional: SimpleBGC GUI

2 Ziel

Das Ziel besteht darin, mithilfe einer Software, Steuerkommandos an den SBG-Controller zu senden, der dann mit den entsprechenden Daten antwortet. Dabei interessieren lediglich die aktuellen Daten (Real Time Data) des Controllers. Die Konfiguration oder ähnliches soll also mit der erwähnten Software weder gelesen noch geändert werden können. Somit wurde die Software dahingehend entwickelt, dass das Steuerkommando periodisch gesendet wird aber immer das Selbe bleibt.

2.1 Software-Konzept

Die Software besteht aus drei Modulen (Klassen). Das erste Modul ist zuständig für die serielle Kommunikation, dazu gehört der Verbsingaufbau der gewählten Schnittstelle, das Lesen sowie das Schreiben von der Schnittstelle. Das zweite Modul dient zum interpretieren der erhaltenen Rohdaten. In erster Linie werden damit die Winkel des Controllers sowie der externen IMU (inertial measurement unit) ausgewertet. Weiter werden natürlich auch die restlichen vom Controller erhaltenen Daten ausgewertet. Dazu gehören unter anderen Fehlermeldungen oder die Batteriespannung. Das dritte Modul, dient momentan lediglich dazu, die gelesenen Daten in ein CSV-File zu schreiben.
Die Klassen und deren Methoden werden dann in der Main-Methode initialisiert und aufgerufen.

2.2 Programmablauf

Der Programmablauf sieht wie folgt aus:

Zyklischer Programmablauf:

- Initialisieren
- Bestimmen der Zykluszeit
- Abfragen der IMU-Daten
- Lesen der Antwort
- Validieren der Daten
- Interpretieren der Daten
- Schreiben ins CSV-File

2.3 Serielle Schnittstelle

Die Klasse „Serial“ dient dazu zuerst den gewünschten Port mit der entsprechenden Konfiguration zu initialisieren. Diese Konfiguration ist momentan fest programmiert. Die Klasse ist jedoch dafür vorbereitet, dass die Konfiguration beispielsweise als Parameter übergeben werden könnte.

Aktuelle Konfiguration:

- Baudrate: 115200
- 8 Data Bits
- No Parity Bit
- 1 Stop Bit
- kurz: 115200/8-N-1

Die Baudrate entspricht dabei dem Standard des SBG-Controllers. Die Baudrate des Controllers kann falls gewünscht über das GUI geändert werden. Dies ist jedoch nicht empfehlenswert.

Weiter werden die erhaltenen Daten nach dem lesen bereits hier validiert. Dabei wird die Header-Checksumme sowie auch die Body-Checksumme überprüft. Stimmen die Daten nicht mit den erwarteten Werten überein, wird eine Fehlermeldung ausgegeben.

2.3.1 Request

Um vom Gimbal-Controller die gewünschten „Real-Time-Daten“ zu erhalten muss zuerst eine entsprechende Anfrage (Request) gesendet werden. Wie die Anfrage zusammengestellt werden kann, kann ebenfalls den Spezifikationen (Protocol Specifications) entnommen werden. Um die aktuellen Winkel- wie auch IMU-Daten zu erhalten setzt sich die Anfrage wie folgt zusammen:

Request:

Index	Name	Hex	Dec/UTF8
[0]	Character:	0x3E	>
[1]	Command ID:	0x17	23
[2]	Data Size:	0x00	0
[3]	Header Checksum:	(Command ID + Data Size)% 256	
[4]	Data:	0x00	0

Die aufgeführten Zeichen können dann im hexadezimalen Format einem „char“-Array übergeben werden, und können so auf den ausgewählten Port geschrieben werden

2.3.2 Response

Wird dem Gimbal-Controller eine Anfrage in korrektem Format gesendet, sendet der Controller eine entsprechende Antwort. Die Antwort ist ebenfalls wieder ein „char“-Array, und setzt sich, falls in korrektem Format, wie folgt zusammen. hjhj

Response:

Index	Name	Hex	Dec/UTF8
[0]	Character:	0x3E	>
[1]	Command ID:	0x17	23
[2]	Data Size:	0x00	0
[3]	Header Checksum:	(Command ID + Data Size)% 256	
[4...66]	Data:		
[67]	Body Checksum:		

Das somit erhalten „‘char“-Array wird sogleich validiert. Falls die verifiziert werden können, sprich das Format mit dem Erwartungswert übereinstimmt, wird das Array der Klasse „SBGC“ übergeben.

2.3.3 Deadlock

Solange auf jede Anfrage eine Antwort des Controllers kommt, funktioniert das Programm fehlerlos. Geht aus irgendeinem Grund jedoch eine Antwort verloren, zum Beispiel weil nicht gesendet oder nicht detektiert wurde, gerät das Programm in einen Deadlock. Um diesen zu verhindern wird der serielle Port neu initialisiert. Um dabei nachvollziehen zu können wie oft die Verbindung abbricht werden drei Nullen in das File „ang.csv“ geschrieben. Danach beginnt der Programmablauf wieder von vorne, das heisst es wird wieder eine Anfrage gesendet. Der Programmablauf weist somit also ein paar Gemeinsamkeiten zum UDP-Protokoll auf, denn dem Programm ist es weitgehend egal ob eine Antwort des Controllers gesendet wird. Kommt vom Controller keine Antwort wird einfach die nächste Anfrage gesendet.

2.4 Interpretation

Die Klasse „SBGC“ wertet die empfangenen Daten aus. Die Daten sind zuerst in einem Array enthalten. Diese Werte können nun mithilfe der Protokollspezifikationen (Protocol Specifications) entsprechend interpretiert und den entsprechenden Parametern zugeordnet werden. Die Werte des Arrays entsprechen dabei jeweils einem Byte. Die Parameter, die in den Spezifikationen als 2 Byte lang bezeichnet werden, setzen sich dabei jeweils aus zwei Werten des Array zusammen. Dabei ist es wichtig, dass jeweils der zweite Wert vor den ersten geschoben (Shift-Operation) wird um den korrekten Wert zu erhalten.

Beispiel:

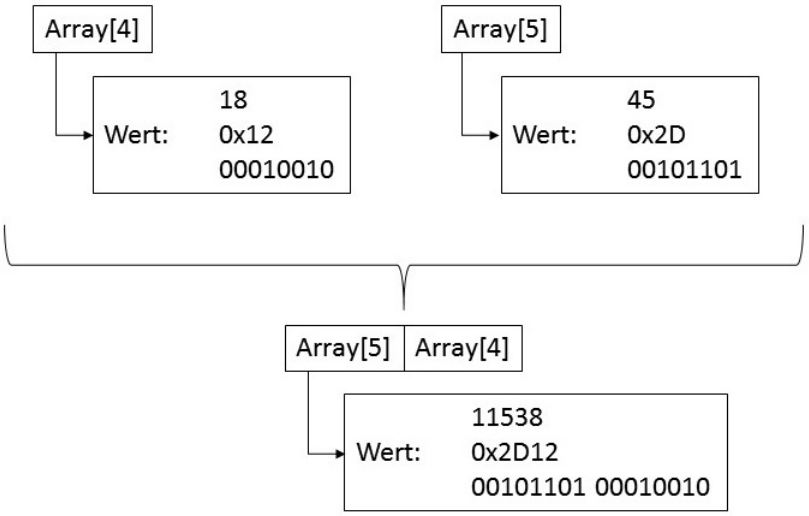

Abbildung 1: Shift-Operation

Wie sich sämtliche Werte genau zusammensetzen kann den Spezifikationen entnommen werden. Schlussendlich werden die jeweiligen Werte in deren korrekten Einheiten den Variablen

eines Structs zugewiesen.

Die wichtigsten Daten sind dabei die Winkel, welche in den Radaralgorithmus einbezogen werden sollen. Die Winkel können dabei von beiden Sensoren (IMUs) gelesen werden, einerseits vom Board (Frame), andererseits von der externen IMU. Da die externe IMU die Winkel relativ zum Boden angibt, sollten in erster Linie diese berücksichtigt werden.

TODO Die Software gibt momentan die Winkeldaten des SBGC auf der Konsole aus, dabei fällt auf, dass diese nicht immer korrekt sind und manchmal stark von den effektiven Werten abweichen. Dieses Verhalten kann jedoch nur beobachtet werden, wenn die Auswertung über UART geschieht. Wird hingegen das USB-Kabel eingesetzt, sind die Werte immer korrekt. Beim Untersuchen der Fehlerquelle zeigte sich, dass der Controller die Anfrage (Request) korrekt beantwortet (siehe S. 3). Die Daten werden somit entweder nicht korrekt in den Buffer aufgenommen oder bei der Verarbeitung geht etwas schief.

Lösungsansatz Das Problem sollte weiter untersucht werden, damit die Fehlerquelle eliminiert werden kann. Alternativ kann durch berechnen der Checksumme überprüft werden welche Daten korrekt sind und welche korrumpiert sind. Somit könnten die Fehlerhaften Daten einfach verworfen werden.

2.5 Write-To-File

Die erhaltenen Daten werden schliesslich in ein CSV-File geschrieben. Dabei werden zwei Files erstellt, eines enthält die Winkel, welche ins Post-Processing einbezogen werden sollen, das andere enthält Statusmeldungen wie zum Beispiel, ob Fehler aufgetreten sind.

Die Winkeldaten werden ausserdem mit einem Zeitstempel versehen. Dieser Zeitstempel stammt vom SBC, und wird während des Schreibprozesses erstellt. Somit stellt sich eine Verzögerung vom Lesen des Gimbal-Controllers bis zum Schreiben des tatsächlichen Zeitstempels ein. Wie gross diese Verzögerung momentan ist, ist nicht bekannt. Sollte sich jedoch herausstellen, dass dies ein Problem ist, muss der Zeitstempel an anderer Stelle im Programm abgerufen werden.

Die restlichen Daten der IMU werden ebenfalls zyklisch in ein File geschrieben, aus Performancegründen werden diese Daten jedoch nur in jedem fünfzigsten Zyklus im CSV-File abgelegt. Dieses File ist vor allem wichtig um sicherzustellen, dass während des Fluges keine Fehler aufgetreten sind.

3 Level-shifter

Der Phytec SBC liefert an der UART-Schnittstelle lediglich 3.3V, der SBGC gibt jedoch nur auf 5V Signale eine Antwort. Deshalb muss einem Level-Shifter das Signal nach oben gezogen werden. Unter Umständen reicht es aus lediglich die Frage (Request auf Rx) von 3.3V auf 5V zu transformieren da Tx auf dem SBGC lediglich 3.3V Spannung aufweist.

TODO Welche Schaltung genau eingesetzt wird muss noch evaluiert werden. Es stehen bereits Muster zur Verfügung. Ebenfalls ist noch klar wie ganze Verkabelung aussehen wird. Wurde der Level-Shifter erfolgreich eruiert und getestet wird die Schaltung zusammen mit dem Schaltregler und dem Batteriewächter auf einer Platine aufgebracht.

Literaturverzeichnis

[1]