Reverse engineering & tool ecosystem for Yuneec drones – a showcase

- Über mich
- Yuneec
- Der Anfang
- Logikanalysator
- Toolsammlung
- Calibration tools

Über mich

Helmut Elsner, 68 Jahre alt, Rentner
h-elsner im Lazarusforum (und andere Foren)

- - 10 Jahre Schule
- - Nachrichtentechniker
- - RADAR Ingenieur
- - SigInt Experte
- - Customer Support für große Telefonvermittlungsstellen
- - Testsupport für Mobilfunk Infrastruktur (eNodeB)

Links

http://h-elsner.mooo.com
https://github.com/h-elsner
https://gitlab.com/h-elsner

Yuneec → ATL

Yuneec (China)

https://yuneec.com

ATL (CH, USA)

https://shop.yuneec.com/

Model name	First flight	Type
EPac		Electric powered paraglider
E430	2009	Aircraft
e-Spyder		Aircraft
ETrike		Aircraft
EViva	2012	Aircraft
Horizon Hobby Blade 350QX		Quadcopter
Horizon Hobby Chroma		Quadcopter
H920 Tornado	2015	Hexacopter
Q500 Typhoon	2015	Quadcopter
Breeze	2016	Quadcopter
Typhoon H	2016	Hexacopter
H520	2017	Hexacopter
Typhoon H Plus	2018	Hexacopter
Mantis Q	2018	Quadcopter
Mantis G		Quadcopter
H520E	2020	Hexacopter
H850	2022	Hexacopter

Meine Kopter

2014 erster Quadkopter: Blade 350QX
2015 einen Yuneec Q500 mit stabilisierter Kamera CGO3

- Da fielen mir die Flight logs auf – Idee: Umwandlung in Flugpfad für GoogleEarth (kml, kmz dateien) → 1. Tool Q500log2kml

Ab dann weiter Hexakopter wie Typhoon H und H920 mit mehr Flugspaß und auch mehr technische Möglichkeiten.

- Fliegerei tritt dann allerdings hinter Technik zurück

Q500log2kml

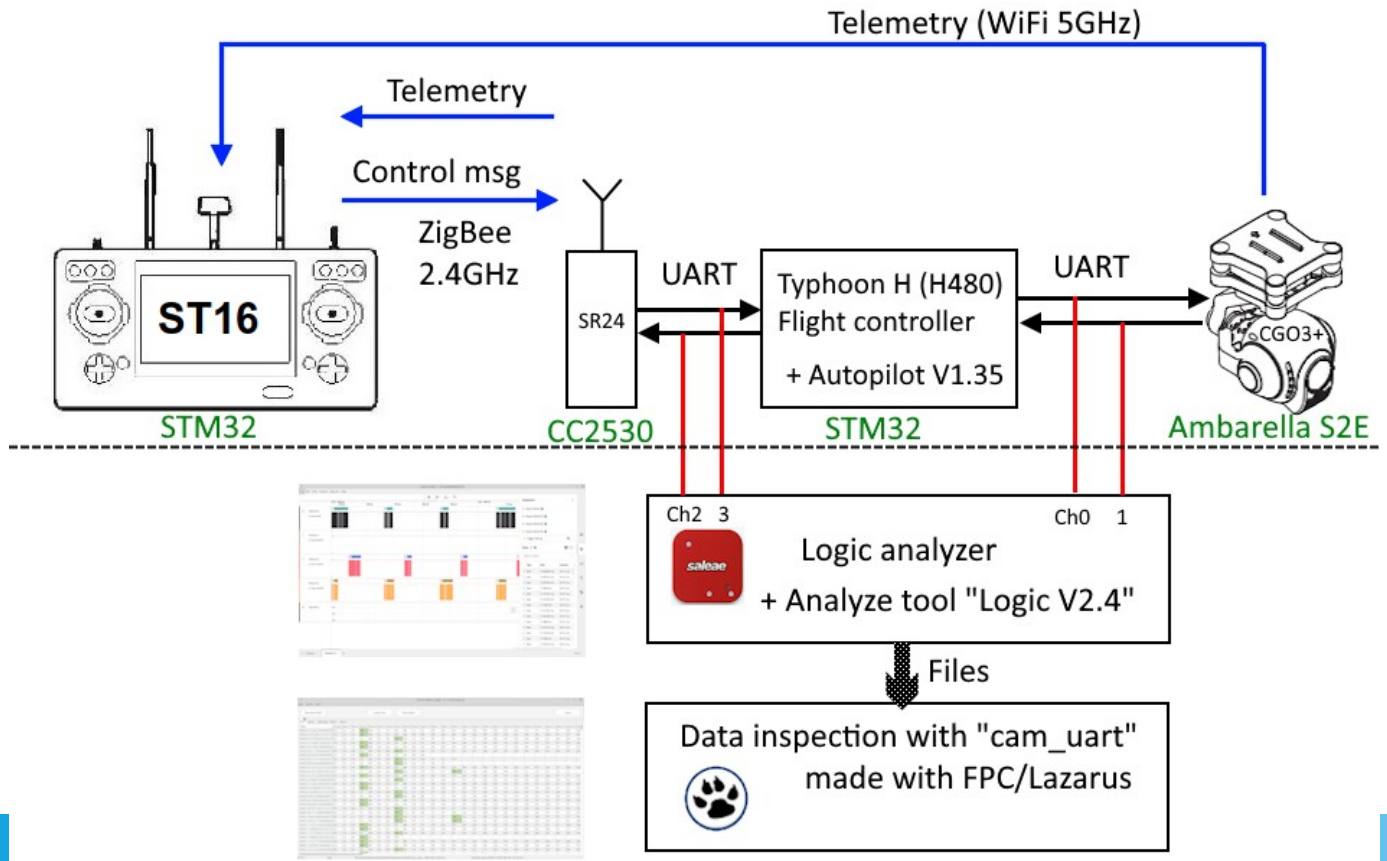
Eines der ersten größeren Projekte mit Lazarus.
Seit 2015 bis heute – von der simplen Idee, die der Name suggeriert zum universellen Analyse tool für Flight logs aller Quad- und Hexakopter von Yuneec – wird gern und weltweit in der Community genutzt.

Quelltext – schönes Beispiel, wie man es nicht machen sollte:
- Alles in einem Programm – komplexe Benutzeroberfläche (CGO3control entfernt)
- Importfilter nicht sauber von Verarbeitung getrennt
- immer etwas zugefügt, je nach aktuellem Fall – zum Funktionen, die ich selber bereits vergessen habe
- Komplexe (zu lange) Procedures und Functions → FPC_Understand
- Aufteilung in Units nicht konsequent ...

Schlussfolgerungen

- Projekt immer so anfangen, als ob man es erweitern will
- Lieber zwei oder mehr Programme als ein Universaltool
- Interfaces (Importfilter), Daten, Darstellung sauber trennen
- Auslagern von Functions und Procedures in eigene Units für universelle Nutzung
- Verwenden von vernünftigen Namen für Variablen, Functions, Procedures ...

Reverse engineering



Tool ecosystem

Wenn man die Kommunikation der einzelnen Systeme (RC, Flight Controller, Gimbal, Camera) kennt, kann man drei Dinge tun:

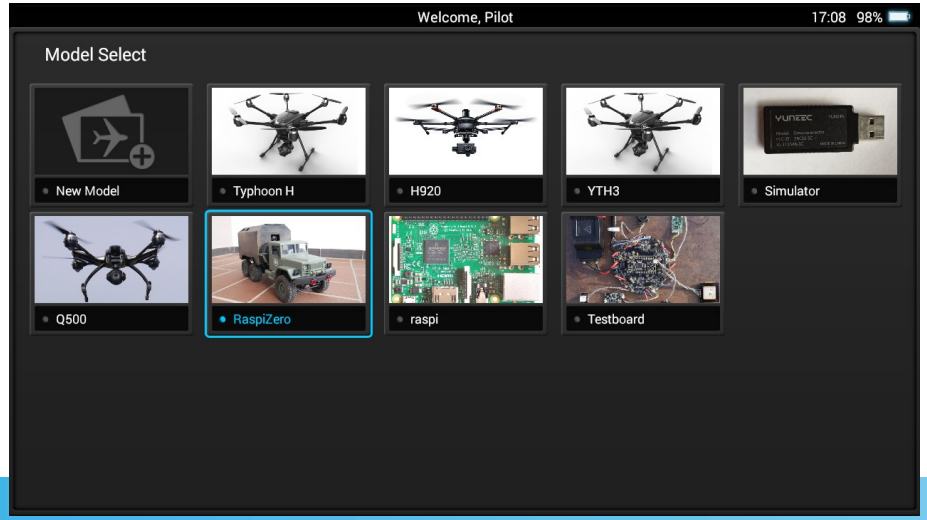
- Tools zur Reparatur oder Analyse erstellen oder nachbauen. Tools für LINUX Betriebssysteme gab es gar nicht.
- Die einzelnen Systeme anderweitig nutzen, wie zum Beispiel die recht umfangreiche Funkfernsteuerung oder die stabilisierte Kamera.
- Systeme die nicht kompatibel sind, kompatibel machen. Zum Beispiel eine Wärmebildkamera von einen neueren Koptertyp an meinem alten System nutzen.

Tool ecosystem

- Drei einzelne Programme für drei Kamerafamilien – CGO3/GB203, CGO3+, C23/E90
- Programm zur Kontrolle der Sensoren des Flight Controllers für Typhoon H (gab es nur für Windows und dieses ist besser).
- Programm zum Steuern der Kameraeinstellungen und Testen der CGI Kommandos
- Programm zum Extrahieren von Einzelkomponenten der Firmware
- Analyse Flight Logs mit kleinem Ableger Batterietest

Umnutzung

- Funkfernsteuerung für andere Systeme nutzen (Auto, Boot…)
- Kamera an anderen Systemen nutzen. Demoprogramme, die zeigen dass und wie es geht.
- Demoprojekt: Fernsteuerung mit Raspberry Pi gekoppelt um ein Fahrzeug zu steuern.



Schluss

Dank an alle, die FPC/Lazarus möglich gemacht haben und für uns weiterentwickeln.

Und nicht zu vergessen die Hilfe im deutschen Lazarusforum.