Thunderbird



... ist eine customized firmware für den Yuneec Typhoon H (manchmal auch als H480 bezeichnet) auf Basis vom PX4 Autopilot, entwickelt von **Toni Rosendahl**. Erstveröffentlichung siehe hier: <u>https://yuneecpilots.com/threads/typhoon-h-480-px4-v1-10-stability-issues.18205/page-3</u>

Die Firmware ist Open Source. Alle können mitmachen. Das Projekt wird hier gepflegt: <u>https://github.com/tonirosendahl/Thunderbird</u> PX4 Autopilot: <u>https://docs.px4.io/</u>

Wie flashe ich ein MCU-Board vom Typhoon H

Achtung: Wenn der Bootloader einmal aufgespielt wurde, gibt es kein zurück mehr. Der Typhoon H ist dann ein PX4 Thunderbird.

Was wir brauchen:

Binaries zum Flashen downloaden und auspacken: https://github.com/tonirosendahl/Thunderbird/blob/Typhoon H 480/Thunderbird 19122019 FT.zip

Einen Progammer für STMxx mit einen **ST-Link V2-1**. So etwas wie zum Beispiel das: <u>https://www.st.com/en/evaluation-tools/nucleo-f767zi.html</u>



Dafür müssen wir noch das ST-Link utility downloaden und installieren. <u>https://www.st.com/en/development-tools/stsw-link004.html</u>

Und natürlich brauchen wir ein MCU-Board vom Typhoon H. Am besten man hat ein Ersatz Board. Dann können wir nämlich durch umstecken eines normalen Boards auf den alten Typhoon H zurückfallen.

Test: Zuerst das Board auf Funktion prüfen. Beim Anschließen muss es hochlaufen, zu erkennen an den LEDs. Im Windows Geräte-Manager erscheint es als "STMicroelectronics Virtual COM Port (COM...)".



Um die Parameter einzustellen und spätere Firmwareupdates zu machen müssen wir "QGroundControl" downloaden und installieren. Starten sollte man das Tool als Administrator (nötig jedenfalls bei Windows 10).

http://qgroundcontrol.com/downloads/

Achtung:

Bevor wir weitermachen, lesen wir erst die Dokumentationen für das Zeug oben. Bitte auch die ESD-Regeln beachten. Alles erden, was geht.

Schritt 1: Den Programmer anschließen (der kritischste Schritt)

Wir brauchen fünf Anschlüsse: Ground (GND), +3.3V (Vcc), clock (SWCLK), data (SWDIO) und reset (NRST). Für alle diese Anschlüsse gibt es kleine, runde Meßpunkte auf dem MCU-Board. Nun ist es an euch, wie ihr diese mit dem SWD-Port des ST-Links verbindet Im einfachsten Fall dünne Drähte anlöten oder, wie ich, kleine Buchsen aus alten IC-Fassungen.



Man kann auch die vorgeleisteten Lötpads für einen Steckverbinder nutzen. Passende Steckverbinder müssen allerdings erst aufgetrieben werden. Eine gute Quelle wäre ein defektes Gimbalboard der CGO3(+). Nur NRST muss dann extra extra noch angeschlossen werden.



Der Steckverbinder sollte mit einem Tropfen Sekundenkleber fixiert werden. Bitte immer Vorsicht beim Stecken und besonders beim Ziehen der Kabel.

Test: Nach dem Löten sollte man das Board erneut auf Funktion prüfen. Beim Anschließen über USB muss es hochlaufen, zu erkennen an den LEDs. Im Windows Geräte-Manager sollte es immer noch als "STMicroelectronics Virtual COM Port (COM...)" erscheinen.

Schritt 2: Den originalen Bootloader durch dem PX4 Bootloader ersetzen (ab jetzt gibt es kein zurück!)

Beide CN4 Jumper A und B müssen offen sein (Brücken entfernt), um den SWD-Port für externe Prozessoren freizugeben.

Verbinde das MCU-Board via USB mit dem PC, um die Stromversorgung sicherzustellen, auch wenn das MCU-Board auf der ESC-Platine steckt.

Verbinde den SWD-Port mit den Messpunkten auf dem MCU-Board.



Wenn alles richtig wie oben gezeigt verbunden ist, starte das STM32 ST-Link Utility.

Menü Target > Connect, um festzustellen, ob das MCU-Board korrekt verbunden ist. Die Device ID und die Device family: STM32F... sollten erkannt werden. Wenn nicht, dann Verkabelung prüfen. Siehe Bilder oben.

Read Out Protection auf Level 0 setzen:

Target > Option Bytes... "Level 0" auswählen > Apply.

Jetzt wird der alte Bootloader unwiderruflich entfernt.

Read Out Protection	B	OR Leve	1	
Level 0	~ F	OFF	~ R	~
User configuration op IWDG_STOP WWDG_SW nSRAM_Parity SRAM2_RST SRAM2_PE nRST_SHDW RST_STOP	tion byte IWDG_STD IWDG_ULP FZ_IWDG_S FZ_IWDG_S PCROP_RD nBoot0_SW nSWB00T0	BY STOP STDBY P Cfg	nBoot0 nBoot1 nDBOOT nDBANK DB1M IRHEN WDG_SW	nBOOTO BOOT1 nBFB2 nBOOT_SEI DUALBANK BOREN
MRSI_SIDBY	VDDA_Mon	itor	SDADC12_VL	DD_Monitor
NRST_MODE			~	
Security option bytes SEC_SIZE	00 SEC	_SIZE2	0x00	BOOT_LOCK
Boot address option t	oytes	Boot from	o (H)	
0001_A000((1)		DOOLIIOI	(i) (i)	
DOOT ADD1 (U)		Doot from		
BOOT_ADD1 (H) User data storage op	tion bytes	Boot from	n(H)	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protecti	tion bytes	Boot fron	ata 1 (H)	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protecti Sector	tion bytes ion Start address	Boot fror Da Size	n (H) ata 1 (H) Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protecti Sector	tion bytes ion Start address 0x08000000	Boot from Da Size 16 K	n (H) ata 1 (H) Protection	^
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protecti Sector Sector 0 Sector 1	tion bytes ion Start address 0x08000000 0x08004000	Boot from Da Size 16 K 16 K	n (H) ata 1 (H) Protection No Protection No Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protecti Sector Sector 0 Sector 1 Sector 2	tion bytes ion Start address 0x08000000 0x08004000 0x08008000	Boot fror Da Size 16 K 16 K 16 K	Protection No Protection No Protection No Protection No Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protecti Sector Sector 0 Sector 1 Sector 2 Sector 3	tion bytes ion Start address 0x08000000 0x08004000 0x08008000 0x08000000	Boot fror Da Size 16 K 16 K 16 K 16 K 16 K	Protection No Protection No Protection No Protection No Protection No Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protecti Sector Sector 0 Sector 1 Sector 2 Sector 3 Sector 4	tion bytes ion Start address 0x08000000 0x08004000 0x08008000 0x08000000 0x08000000 0x08000000 0x08000000	Boot fror Da Size 16 K 16 K 16 K 16 K 16 K 64 K	ata 1 (H) Protection No Protection No Protection No Protection No Protection No Protection No Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protecti Sector Sector 0 Sector 1 Sector 1 Sector 2 Sector 3 Sector 4 Sector 5	tion bytes ion Start address 0x08000000 0x08004000 0x08008000 0x08008000 0x08002000 0x08010000 0x08020000	Boot fror Da Size 16 K 16 K 16 K 16 K 16 K 16 K 16 K 128 K	Protection No Protection No Protection No Protection No Protection No Protection No Protection No Protection No Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protecti Sector Sector 0 Sector 1 Sector 2 Sector 2 Sector 3 Sector 4 Sector 5 Sector 5	tion bytes ion Start address 0x08000000 0x08004000 0x08008000 0x08002000 0x08010000 0x08020000 0x08020000 0x08040000	Boot fror Da Size 16 K 16 K 16 K 16 K 16 K 128 K 128 K	n (H) ata 1 (H) Protection No Protection No Protection No Protection No Protection No Protection No Protection No Protection No Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protecti Sector Sector 0 Sector 1 Sector 2 Sector 2 Sector 3 Sector 4 Sector 5 Sector 5 Sector 6 Sector 7	tion bytes ion Start address 0x08000000 0x08004000 0x08008000 0x0800C000 0x08010000 0x08010000 0x08020000 0x08040000 0x08060000	Boot fror Da Size 16 K 16 K 16 K 16 K 16 K 128 K 128 K 128 K	ata 1 (H) Protection No Protection No Protection No Protection No Protection No Protection No Protection No Protection No Protection No Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protection Sector Sector 0 Sector 1 Sector 2 Sector 2 Sector 3 Sector 3 Sector 4 Sector 5 Sector 5 Sector 6 Sector 7 Sector 8	tion bytes ion Start address 0x08000000 0x08004000 0x08008000 0x08000000 0x08010000 0x08010000 0x08020000 0x08020000 0x08040000 0x08060000 0x08080000	Boot fror Da Size 16 K 16 K 16 K 16 K 16 K 16 K 128 K 128 K 128 K 128 K	Area 1 (H) Protection No Protection No Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protect Sector Sector 0 Sector 1 Sector 2 Sector 2 Sector 3 Sector 3 Sector 4 Sector 5 Sector 5 Sector 7 Sector 7 Sector 8 Sector 9	tion bytes ion Start address 0x08000000 0x08004000 0x08008000 0x08008000 0x08010000 0x08010000 0x08040000 0x08040000 0x08040000 0x08040000	Boot fror Da Size 16 K 16 K 16 K 16 K 16 K 128 K 128 K 128 K 128 K 128 K	A (H) Ata 1 (H) Protection No Protection No Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protecti Sector Sector 0 Sector 1 Sector 2 Sector 2 Sector 3 Sector 4 Sector 5 Sector 5 Sector 6 Sector 7 Sector 8 Sector 9 Sector 9	tion bytes ion Start address 0x08000000 0x08004000 0x08000000 0x08000000 0x08010000 0x08020000 0x08040000 0x08060000 0x08060000 0x08060000 0x08060000 0x08060000 0x08060000	Boot fror Da Size 16 K 16 K 16 K 16 K 16 K 128 K 128 K 128 K 128 K 128 K 128 K	ata 1 (H) Protection No Protection No Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protection Sector 0 Sector 1 Sector 2 Sector 2 Sector 3 Sector 4 Sector 5 Sector 5 Sector 5 Sector 6 Sector 7 Sector 8 Sector 9 Sector 10 <	tion bytes ion Start address 0x08000000 0x08004000 0x08008000 0x08000000 0x08010000 0x08010000 0x08010000 0x08040000 0x08060000 0x08060000 0x08060000 0x08060000 0x08060000	Boot fror Da Size 16 K 16 K 16 K 16 K 16 K 128 K 128 K 128 K 128 K 128 K 128 K 128 K	ata 1 (H) Protection No Protection No Protection	
BOOT_ADD1 (H) User data storage op Data 0 (H) Flash sectors protection Sector 0 Sector 1 Sector 2 Sector 2 Sector 3 Sector 4 Sector 5 Sector 5 Sector 5 Sector 6 Sector 7 Sector 8 Sector 9 Sector 9 Sector 10 <	tion bytes ion Start address 0x08000000 0x08004000 0x08000000 0x08000000 0x08010000 0x08010000 0x08020000 0x08040000 0x08060000 0x08060000 0x08060000 0x08060000 0x08060000 0x08060000	Boot fror Da Size 16 K 16 K 16 K 16 K 16 K 128 K 128 K 128 K 128 K 128 K 128 K	ata 1 (H) Protection No Protection No Protection	

Das sieht dann so aus:

遍 STM32 ST-LIN	K Utility						-		×
<u>F</u> ile <u>E</u> dit <u>V</u> iew	<u>T</u> arget ST-L	INK External L	oader <u>H</u> elp						
🖴 🖥 🖕	Ç 🏈 🥋	š 😥 🔜							
Memory display						Device	STM32F405xx/F407xx/F415xx/F4	17xx	
Address: 0x080	000000 V Siz	e: 0x1000	Data W	idth: 32 bits	~	Device ID	0x413		
						Revision ID	Rev 2.0		
Device Memory @	0x08000000 : j	Binary File				Fidari aize	UNKNOWN	LiveU	pdate
Target memory, Add	lress range: [0x0	08000000 0x0800	1000]						
Address	0	4	8	С	ASCII				^
0x08000000	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿ	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	ÿÿÿ		
0x08000010	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿ	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	ÿÿÿ		
0x08000020	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿ	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	ÿÿÿ		
0x08000030	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿ	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	ÿÿÿ		
0x08000040	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿ	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	ÿÿÿ		
0x08000050	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿ	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	ÿÿÿ		
0x08000060	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿ	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	ÿÿÿ		
0x08000070	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿ	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	ÿÿÿ		
0x08000080	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿ	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	ÿÿÿ		~
<									>
13:09:50 : V232501 13:09:50 : Connect 13:09:50 : SWD Ere	ed via SWD.								^
13:09:50 : Connect	ion mode : Norm	al.							
13:09:50 : Debug in 13:09:50 : Device If	Low Power mod D:0x413	le enabled.							
13:09:50 : Device f	amily :STM32F40	15xx/F407xx/F415	ixx/F417xx						
Disable F	Read Out Protect	tion and retry.							- 64
13:10:27 : Option b	ytes updated su	ccessfully.							~
Debug in Low Power	mode enabled.		Device ID:0x	(413			Core State : Live Update Disabled		

Nun flashen wir den neuen Bootloader auf Adresse 0x08000000. Menü Target > Program & Verify. Die Datei "**omnibusf4sd_bl.bin**" aus den Binaries auswählen > Start.

Download [omr	nibusf4sd_bl.bin]	×
Start address	0x08000000	
File path Extra options	D:\Yuneec_Kopter\Thunderbird	d\Flashing_YTH_MCU\Thi Browse
	🗌 Skip Flash Erase	Skip Flash Protection verification
Verification	Verify while programming	O Verify after programming
Click "Start" to p	program target.	
After programn	ning ☑ Reset after programming	E Full Flash memory Checksum
	Start	Cancel

Nun sollte es so aussehen:

STM32 ST-LINK	Utility						_	-		×
<u>File Edit V</u> iew	Target ST-LIN	IK External Lo	ader <u>H</u> elp							
🖴 🖥 🛛 🗳 <	Ç 🤣 💱	🧼 駴								
Memory display						Device	STM32F405xx/F407xx/F415x	x/F417	xx	
Address: 0x0800	00000 V Size:	: 0x2690	Data Widt	h: 32 bits 🗸		Device ID	0x413			
						Revision ID	Rev 2.0			
Device Memory @ 0	x08000000 : Fil	e : omnibusf4sd	bl.bin			Flash size	Unknown	Г	Livell	odata
Target memory, Addr	ess range: [0x08	000000 0x08002	2690]					L		Juate
Address	0	4	8	С	ASCII					^
0x08000000	20020000	08001BF5	08001BF3	08001BF1	õ.	óñ				_
0x08000010	08001BF1	08001BF1	08001BF1	00000000	ññ.	ñ				_
0x08000020	00000000	00000000	00000000	08001BF3		ó				_
0x08000030	08001BF3	00000000	08001BF3	080002D1	ó	óÑ				_
0x08000040	08001BF1	08001BF1	08001BF1	08001BF1	ññ.	ññ				_
0x08000050	08001BF1	08001BF1	08001BF1	08001BF1	ññ.	ññ				_
0x08000060	08001BF1	08001BF1	08001BF1	08001BF1	ññ.	ññ				_
0x08000070	08001BF1	08001BF1	08001BF1	08001BF1	ññ.	ññ				_
0x08000080	08001BF1	08001BF1	08001BF1	08001BF1	ññ.	ññ				
<	1	1	i	1	1					>
13.05.30 . Device 10			a/E417bar						_	~
13:09:51 : Can not re	ead memory!	xx/F407xx/F415	(X/F41/XX							
Disable Re	ead Out Protectio	n and retry.								
13: 10: 27 : Option bytes updated successfully. 13: 13:01 : [omnibusf4sd_bl.bin] opened successfully.										
13:13:01 : [omnibusf	13:13:01 : [omnibusf4sd_bl.bin] checksum : 0x000E5189									
13:14:21 : Memory programmed in Us and 563ms. 13:14:21 : VerificationOK										
13:14:21 : Programm	13:14:21 : Programmed memory Checksum: 0x000E5189									
										*
Debug in Low Power r	ebug in Low Power mode enabled. Device ID:0x413 Core State : Live Update Disabled									

Das MCU-Board hat nun den PX4 Bootloader am Anfang des Flashbereichs, 0x8000000. Das ist die Adresse, wo der Bootprozess startet. Man kann hier nichts kaputt machen. Was immer hier ist, wird vom Bootloader gestartet, der fest im ROM eingebrannt ist. Dem ist es egal, was sich hier befindet. Ist etwas falsch gelaufen, kann man einfach erneut flashen.

Die derzeitige PX4 Firmware, genauer gesagt der PX4 Autopilot, startet immer von Adresse 0x8008000. Damit ist genug Platz, um Parameter zwischen Bootloader und PX4 Hauptanwendung zu speichern. Solange wir keine SD-Karte angeschlossen haben, brauchen wir den Platz. Man könnte die PX 4 Anwendung auch gleich auf Adresse 0x8000000 starten und bräuchten den Bootlader gar nicht, aber dann müssen die Parameter auf einer (hier nicht vorhandenen) SD-Karte gespeichert werden.

Bis jetzt werden wir weiter mit dem Bootloader arbeiten und nichts ändern.

Schritt 3: Die PX 4 Hauptanwendung flashen

Es gibt zwei Möglichkeiten, die Hauptanwendung nämlich der PX4 Autopiloten zu flashen:

A) Mit Hilfe des Bootloaders: Der gerade aufgespielte PX4 Bootloader kann die Hauptanwendung via Typhoon H USB Buchse flashen und von Adresse 0x8008000 starten. Wir brauchen den ST-Link gar nicht mehr und könnten den Typhoon H zusammenbauen.

Der Bootloader braucht ein paar Sekunden zum flashen. Das Flashen wird mit einem etwas abgewandeltem px_uploader.py Script gemacht. Dieses Script erfordert, die *.bin Datei in eine *.fw zu konvertieren. Schau dir das "flash_typhoon_bootloader.py" Script an und trage dort den COM-Port ein, der der USB-Verbindung zum Typhoon H zugeordnet ist.

Alle genannten Dateien sind in den Binaries enthalten.

B) Weil wir unser MCU-Board noch so schön angeschlossen haben, nehmen wir ST-Link auch gleich zum Flashen des PX Autopilot. Einfach die Datei "yuneec_typhoon_h.bin" genau wie den Bootloader vorhin auf das MCU-Board flashen, aber dieses Mal ab Adresse 0x8008000.

Option A) ist für spätere FW-Updates interessant. Option B) ist genau richtig für jetzt.

Menü Target > Program & Verify. Datei "**yuneec_typhoon_h.bin**" aus den Binaries auswählen. Nach dem Auswählen der Datei, Start address auf 0x8008000 setzen und Start drücken.

Download [yun	eec_typhoon_h.bin]	×
Start address	0×08008000	
File path	D:\Yuneec_Kopter\Thunderbir	rd\Flashing_YTH_MCU\Thu Browse
Extra options	yuneed	_typhoon_h.bin
	📃 Skip Flash Erase	Skip Flash Protection verification
Verification	 Verify while programming 	◯ Verify after programming
Click "Start" to p	program target.	
After programm	ning 🗹 Reset after programming	Full Flash memory Checksum
	Start	Cancel

Nun sollte es so aussehen:

遍 STM32 ST-LINK	Utility						- C) ×
<u>F</u> ile <u>E</u> dit <u>V</u> iew	Target ST-LI	NK External Lo	ader <u>H</u> elp					
🖴 🖥 🛯 🗳 «	Ç 🤣 💱	🧟 🐼						
Memory display						Device	STM32F405xx/F407xx/F415xx/F417xx	
Address: 0x0800	08000 🗸 Size	: 0xF0404	Data Wid	th: 32 bits 🗸	1	Device ID	0x413	
]	Revision ID	Rev 2.0	
Device Memory @ 0:	x08008000: F	ile : yuneec_typh	oon_h.bin			Flash size		iveUpdate
Target memory, Addr	ress range: [0x08	8008000 0x080F8	3404]					
Address	0	4	8	С	ASCII			^
0x08008000	200053E2	080081B1	080086D9	080086D9	âS. ±	Ù+Ù+		
0x08008010	080086D9	080086D9	080086D9	080086D9	Ù+Ù	+Ù+Ù+.		
0x08008020	080086D9	080086D9	080086D9	080086D9	Ù+Ù	+Ù+Ù+.		
0x08008030	080086D9	080086D9	080086D9	080086D9	Ù+Ù	+Ù+Ù+.		
0x08008040	080086D9	080086D9	080086D9	080086D9	Ù+Ù	+Ù+Ù+.		
0x08008050	080086D9	080086D9	080086D9	080086D9	Ù+Ù	+Ù+Ù+.		
0x08008060	080086D9	080086D9	080086D9	080086D9	Ù+Ù	+Ù+Ù+.		
0x08008070	080086D9	080086D9	080086D9	080086D9	Ù+Ù	+Ù+Ù+.		
0x08008080	080086D9	080086D9	080086D9	080086D9	Ù+Ù	+Ù+Ù+.		
<	1		i					>
13,13,01, [0000003	HSU_DRUNI UPER	ieu successiuiry.	00					^
13:14:21 : Memory p	programmed in 0s	and 563ms.	09					
13:14:21 : Verificatio	nOK	rksum: 0x000E51	89					
13:18:18 : [yuneec_	typhoon_h.bin] o	opened successfu	lly.					
13:18:18 : [yuneec_ 13:20:05 : Memory p	typhoon_h.bin] (programmed in 24	checksum : 0x066 Is and 687ms.	F205A					
13:20:05 : VerificationOK								
13:20:05 : Programm	ned memory Che	cksum: 0x066F20	5A					~
Debug in Low Power r	mode enabled.		Device ID:0x4	13			Core State : Live Update Disabled	

Erledigt!

Alle USB Verbindungen trennen und abschalten. Die Programmer-Verkabelung zum MCU-Board trennen. ST-Link wird nicht mehr gebraucht, es sei denn aus irgendwelchen Gründen wird ein neuer Bootloader gebraucht. Das ist unwahrscheinlich, kann aber in so einer frühen Phase der Projektentwicklung nicht ausgeschlossen werden.

Test:

MCU-Board wieder über USB mit dem PC verbinden. Die LEDs auf dem MCU-Board haben keine Funktion mehr, aber im Windows Geräte Manager > Anschlüsse (COM & LPT) mus jetzt ein Gerät "Legacy FMU (COM...)" auftauchen. Wenn das so ist, dann funktioniert es wie erwartet..

Den Thunderbird zusammenbauen und neu starten. Der Powerbutton hat jetzt leider eine Verzögerung von rund 8 Sekunden. Der Powerbutton muss gedrückt gehalten werden bis die LEDs angehen. Ausschalten geht nur durch Entfernen der Batterie.

Schritt 4: Parameter einstellen

Wir müssen QGroundControl (QGC) installieren und als Administrator starten. Kopter einschalten und über USB mit dem PC verbinden.

QGroundControl	v3.5.6							×	D X
Datei Widget									
ي 🌺 🕲	a 🖪	🦷 🔏 💩 🗉	100% Manual Ents	chärft					
Vehicle Setup	Eine oder mehrere Fluggerätekomponenten müssen vor dem Flug eingestellt werden.								
Summary	Flug	geralelyp 🔷	301150	e ا				Flugmodi	•
Firmware	System ID Fluggerätetyp Fahrzeug Gene	1 Hexarotor x eric Hexarotor x geometry	Compass 0 Gyro Accelerometer	Setup required Setup required Setup required	Roll Pitch Yaw	Setup required Setup required Setup required	Modus Schalter Flug Modus 1 Flug Modus 2	Einrichtu	ng erfoderlich Unassigned Unassigned
Fluggerätetyp	rimware version	1.10.0dev			Aux1 Aux2	Disabled Disabled	Flug Modus 3 Flug Modus 4 Flug Modus 5 Flug Modus 6		Unassigned Unassigned Unassigned
((e)) Sensors									ondusigned
Radio									
Flugmodi	F	Power 🗧	Sicherh	eit					
Power	Battery Full Battery Empty Number of Cells	4.05 V 3.50 V 0	Low Battery Failsafe RC Loss Failsafe RC Loss Timeout Data Link Loss Failsafe	Warning Return mode 0.5 s Disabled					
Sicherheit			RTL Climb To RTL, Then	30.0 m Land immediately					
Tuning									
Parameters									
Parameters									

QGC erkennt automatisch, dass Parameter fehlen und eingestellt werden müssen. Am besten wir laden sie aus der mitgelieferten Datei "**typhoon_h_parameters.params**". QGC > Parameters > Tools > Load from File...

QGroundControl	v3.5.6				- 0	×
Datei Widget						
ي 🎭 🕲	🖉 🗟 🔺	🔏 📥 🗐 1	.00% Manual Entsch	ärft	E	1X4
Vehicle Setup	Search:	Clear				Tools
Summary	Standard	BAT_ADC_CHANNEL		Battery ADC Channel		Refr
Firmware	Battery Calibration	BAT_A_PER_V		Battery current per volt (A/V)		Res
	Commander	BAT_CAPACITY	-1 mAh	Battery capacity	Load from file	• Loa
Fluggerätetyp	Data Link Loss	BAT_CNT_V_CURR		Scaling from ADC counts to volt on the ADC input (battery current)		Save
(tal) Sensors		BAT_CNT_V_VOLT		Scaling from ADC counts to volt on the ADC input (battery voltage)		Clea
() Sensors	EKF2	BAT_CRIT_THR	7 %	Critical threshold		Reb
Radio	Events	BAT_EMERGEN_THR	5 %	Emergency threshold	_	
BOL Flugmodi	Failure Detector	BAT_LOW_THR	15 %	Low threshold		
	Follow target	BAT_N_CELLS	Unconfigured	Number of cells		
Power	GPS	BAT_R_INTERNAL	-1.000 Ohms	Explicitly defines the per cell internal resistance		
Sicherheit	GPS Failure Navigation	BAT_SOURCE	Power Module	Battery monitoring source		
Balancien	Geofence	BAT_V_CHARGED	4.05 V	Full cell voltage (5C load)		
입성 Tuning		BAT_V_DIV		Battery voltage divider (V divider)		
-	Land Detector	BAT_V_EMPTY	3.50 V	Empty cell voltage (5C load)		
Parameters	MAVLink	BAT_V_LOAD_DROP	0.30 V	Voltage drop per cell on full throttle		
	Mission	BAT_V_OFFS_CURR	0.0000000	Offset in volt as seen by the ADC input of the current sensor		
	Mount					
	Multicopter Attitude Control					

Schritt 5: Hardware einstellen und Sensoren kalibrieren

Für Typhoon H gibt es zwei unterschiedliche Kompass Chips. Ältere GPS Module haben einen HMC5883 verbaut, neuere Module haben einen IST8310 als Kompass Chip. Um herauszufinden, welchen du hast, schließe den Kopter an QGC an und gehe zur Mavlink Konsole:

Eingabe:

```
cd /dev
ls
```

und eine Liste der Hardware-Treiber erscheint.

Schön, was wir hier alles sehen können! Für den Kompass gibt es zwei Möglichkeiten: "hmc5883_ext" oder "ist8310_ext".



Wichtig: Wenn man "hmc5883_ext" hat, muss man die External Compass Orientation auf "ROTATION_YAW_270" einstellen.

"ist8310_ext" muss " ROTATION_YAW_180" haben.

Niemals die Autopilot Orientation ändern. Hier muss immer "ROTATION_NONE" stehen.

Gehe zu Settings > Sensors > Set Orientation

0	GroundContro	ol v3.5.6
File	Widgets	
٩	%	🞗 🕢 📴 🕂 📢 🕺 🕺 💩 🛲 🗐 50% Manual Disarmed
Ve	ehicle Setup	
4	Summary	Sensors Setup is used to calibrate the sensors within your vehicle.
		Compass 🕒
	Firmware	Gyroscope Start the individual calibration steps by clicking one of the buttons to the left.
•	Joystick	Accelerometer
	Airframe	
((•))	Sensors —	Cancel
00	Radio	Set Orientations

Set Orientations	Ok
Set your compass orientations below ar the make sure to reboot the vehicle prio flight.	nd or to
Reboot Vehicle	
If the orientation is in the direction of f select ROTATION_NONE.	flight,
Autopilot Orientation:	
ROTATION_NONE -	
External Compass Orientation:	
ROTATION_YAW_180 -	

Dann bei External Compass Orientation den richtigen Wert abhängig vom Kompass-Chip auswählen. Danach neu booten. Nun müssen noch alle Sensoren kalibriert werden, wie bei einem völlig neuen PX4 Kopter üblich. Siehe dazu PX4 Basic Configuration \cdot PX4 v1.9.0 User Guide. **Niemals die ESCs kalibrieren. Dies ist hier nicht nötig und geht auch nicht**.

https://docs.px4.io/master/en/config/

Abschließend, prüfe Parametereinstellungen, Sensoren und Kalibrierungen.

Handling

Jetzt haben wir einen komplett neuen Kopter mit neuen Flugeigenschaften. Wir müssen erst einmal lernen, den zu fliegen. Dazu bitte genug Freiraum einplanen, weg von Bäumen, Mensch und Tier.

In den mitgelieferten Parametereinstellungen ist die obere Position des Flightmode-Schalters "Manual/Stabilized" (Typhoon ohne GPS), mittlere Stellung ist "Position" (Typhoon H mit GPS, Angle Mode) und die untere ist RTH. Ander Flight Modes wie Acro/Rattitude sind in der Standard-Einstellung nicht vorhanden. Man kann die gewünschte Flight Modes, wie so viele andere Parameter auch, frei einstellen, aber sollte damit vorsichtig sein.

Der rote Startknopf geht nicht mehr. Bitte findet selbst heraus, wie alles bei PX4 funktioniert. Dies soll euch anregen, die Dokumentation zu lesen, denn ihr habt ja jetzt einen völlig neuen Koper. <u>https://docs.px4.io/master/en/flying/</u>

Eines sollte man jedoch noch wissen: Motor-Start/Stop geht nur, wenn der Geschwindigkeitsregler an der rechten Seite auf Hase (Schnell) steht.

Die ST16 bleibt erst einmal unverändert. Die Werkseinstellungen sind OK.

Fehlerbehandlung

- Nach dem Löten der Verbindungen auf den Messpunkten am MCB-Board und vor dem Flashen prüfen, ob die Firmware noch hoch läuft bzw. Kopter fehlerfrei bootet. Wenn nicht, auweia! Bitte mit der Lupe auf Lötbrücken prüfen.

- Wenn der Kopter nach dem Flashen nicht als USB-Gerät erscheint, dann ist der Bootloader beschädigt. In dem Falle muss alles von Anfang an neu geflashed werden.

- Wenn der Kopter als USB-Gerät erscheint, aber nicht hoch läuft, auch wenn der Startbutton länger als 8 Sekunden gedrückt gehalten wurde, dann ist die Hauptanwendung beschädigt. Bitte diese ab Adresse 0x8008000 neu flashen (Schritt 3). Achtung! Die Adresse muss **nach** dem Auswählen der Firmware-Datei eingestellt werden, sie springt sonst wieder zurück.

- Prüfen, ob das MCU-Board richtig steckt. Es kann sein, dass es um einen Pin verschoben aufgesteckt ist. Außerdem neigen die Pins zum Verbiegen. HierUSB hilft eine Lupe.

- Nach jeder HW-Änderung oder Reparatur muss Schritt 5: Hardware einstellen und Sensoren kalibrieren erneut durchgeführt werden.