Reverse engineering serial connections at Yuneec Typhoon H

It's interesting to understand the message format of the communication between remote controller ST16 and drone H480 as well as the communication from flight controller to gimbal/camera CGO3+. This may help to create or improve projects to use camera or remote controller for other systems and further development of Thunderbird firmware (see here).
A first setup using a Raspberry Pi is described here: https://github.com/h-elsner/SR24_decode

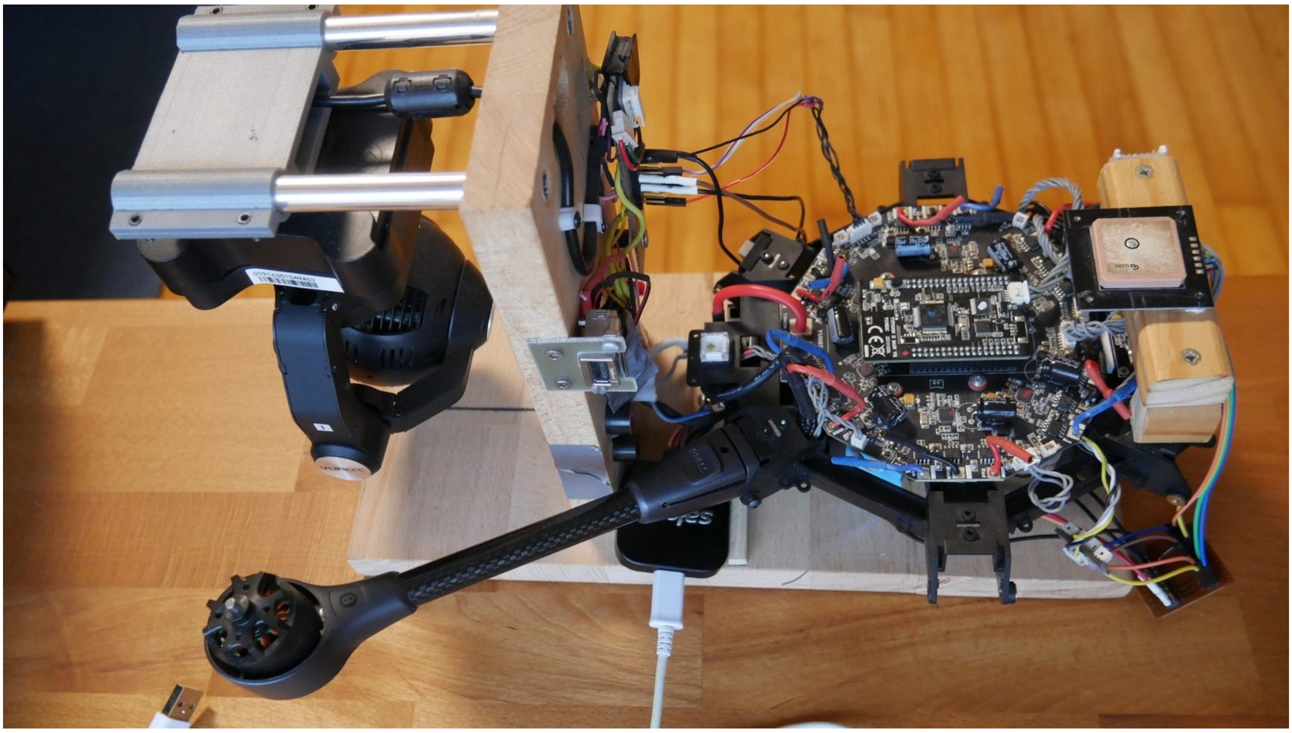
Now I tried to bring together the communication of

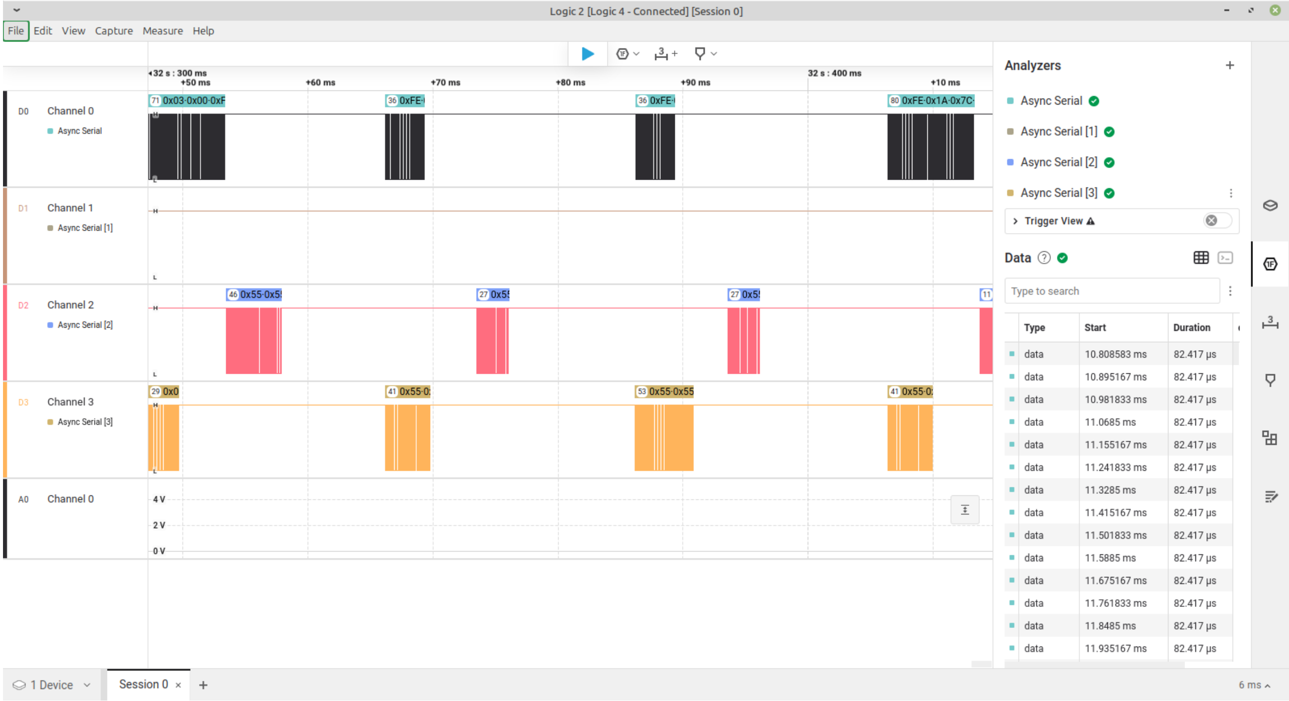
- ST16 <-> SR24 and
- Autopilot (FC = Flight controller) <-> CGO3+.

I got a Saleae Logic Analyzer with four channels. To record I use this analyzer software: https://www.saleae.com/pages/downloads

The setup is as following (with color coding of my wiring to reproduce this setup):

Analyzer channel	connected to	Analyzer color	Cam color	Adapter color	SR24 color	Adapter color
Channel 0	Cam Rx	black	purple/green	grey		
Channel 1	Cam Tx	brown	grey/blue	purple		
Channel 2	SR24 Rx	red			yellow	blue
Channel 3	SR24 TX	orange			orange	green



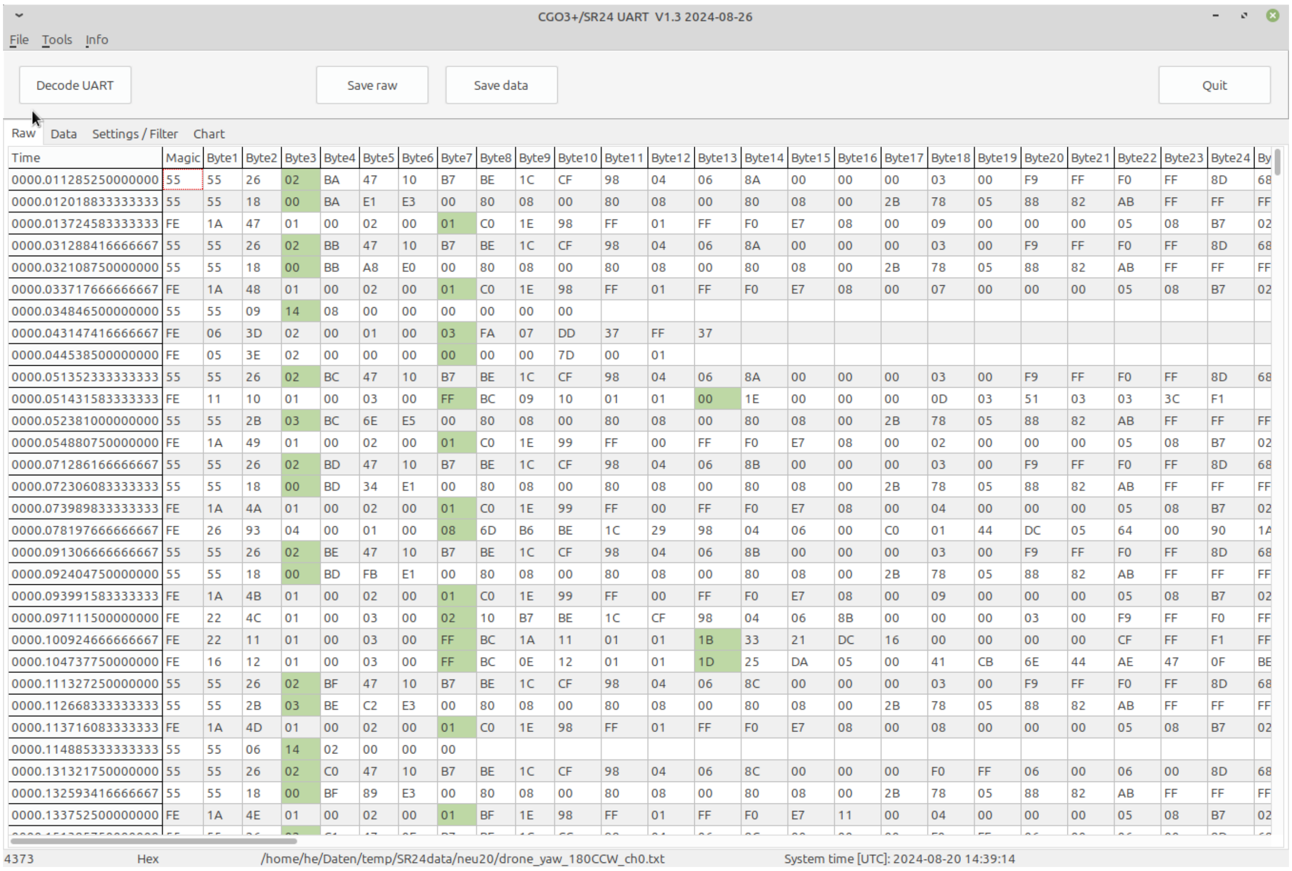


Type	Start	Duration
data	10.808583 ms	82.417 µs
data	10.895167 ms	82.417 µs
data	10.981833 ms	82.417 µs
data	11.0685 ms	82.417 µs
data	11.155167 ms	82.417 µs
data	11.241833 ms	82.417 µs
data	11.3285 ms	82.417 µs
data	11.415167 ms	82.417 µs
data	11.501833 ms	82.417 µs
data	11.5885 ms	82.417 µs
data	11.675167 ms	82.417 µs
data	11.761833 ms	82.417 µs
data	11.8485 ms	82.417 µs
data	11.935167 ms	82.417 µs

The Logic Analyzer records Async Serial in following format:

```
Time [s],Value,Parity Error,Framing Error
0.001172500000000,0xFE,,
0.001259166666667,0x1A,,
0.001345833333333,0x20,,
  etc ...
```

The converter tool "cam_uart" converts this to messages from camera or SR24 which can be stored as CSV files and in some cases can decode it. The CSV files can be checked in any chart tool like Libre Office Calc or Excel.



CGO3+/SR24 UART V1.3 2024-08-26
File Tools Info
Decode UART
Save raw
Save data
Quit
Raw Data Settings / Filter Chart

Time	Magic	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9	Byte10	Byte11	Byte12	Byte13	Byte14	Byte15	Byte16	Byte17	Byte18	Byte19	Byte20	Byte21	Byte22	Byte23	Byte24	By
0000.011285250000000	55	55	26	02	BA	47	10	B7	BE	1C	CF	98	04	06	8A	00	00	00	03	00	F9	FF	F0	FF	8D	68
0000.012018833333333	55	55	18	00	BA	E1	E3	00	80	08	00	80	08	00	80	08	00	2B	78	05	88	82	AB	FF	FF	FF
0000.013724583333333	FE	1A	47	01	00	02	00	01	C0	1E	98	FF	01	FF	F0	E7	08	00	09	00	00	00	05	08	B7	02
0000.031288416666667	55	55	26	02	BB	47	10	B7	BE	1C	CF	98	04	06	8A	00	00	00	03	00	F9	FF	F0	FF	8D	68
0000.032108750000000	55	55	18	00	BB	A8	E0	00	80	08	00	80	08	00	80	08	00	2B	78	05	88	82	AB	FF	FF	FF
0000.033717666666667	FE	1A	48	01	00	02	00	01	C0	1E	98	FF	01	FF	F0	E7	08	00	07	00	00	00	05	08	B7	02
0000.034846500000000	55	55	09	14	08	00	00	00	00	00	00															
0000.043147416666667	FE	06	3D	02	00	01	00	03	FA	07	DD	37	FF	37												
0000.044538500000000	FE	05	3E	02	00	00	00	00	00	00	7D	00	01													
0000.051352333333333	55	55	26	02	BC	47	10	B7	BE	1C	CF	98	04	06	8A	00	00	00	03	00	F9	FF	F0	FF	8D	68
0000.051431583333333	FE	11	10	01	00	03	00	FF	BC	09	10	01	01	00	1E	00	00	00	0D	03	51	03	03	3C	F1	
0000.052381000000000	55	55	2B	03	BC	6E	E5	00	80	08	00	80	08	00	80	08	00	2B	78	05	88	82	AB	FF	FF	FF
0000.054880750000000	FE	1A	49	01	00	02	00	01	C0	1E	99	FF	00	FF	F0	E7	08	00	02	00	00	00	05	08	B7	02
0000.071286166666667	55	55	26	02	BD	47	10	B7	BE	1C	CF	98	04	06	8B	00	00	00	03	00	F9	FF	F0	FF	8D	68
0000.072306083333333	55	55	18	00	BD	34	E1	00	80	08	00	80	08	00	80	08	00	2B	78	05	88	82	AB	FF	FF	FF
0000.073989833333333	FE	1A	4A	01	00	02	00	01	C0	1E	99	FF	00	FF	F0	E7	08	00	04	00	00	00	05	08	B7	02
0000.078197666666667	FE	26	93	04	00	01	00	08	6D	B6	BE	1C	29	98	04	06	00	C0	01	44	DC	05	64	00	90	1A
0000.091306666666667	55	55	26	02	BE	47	10	B7	BE	1C	CF	98	04	06	8B	00	00	00	03	00	F9	FF	F0	FF	8D	68
0000.092404750000000	55	55	18	00	BD	FB	E1	00	80	08	00	80	08	00	80	08	00	2B	78	05	88	82	AB	FF	FF	FF
0000.093991583333333	FE	1A	4B	01	00	02	00	01	C0	1E	99	FF	00	FF	F0	E7	08	00	09	00	00	00	05	08	B7	02
0000.097111500000000	FE	22	4C	01	00	03	00	02	10	B7	BE	1C	CF	98	04	06	8B	00	00	00	03	00	F9	FF	F0	FF
0000.100924666666667	FE	22	11	01	00	03	00	FF	BC	1A	11	01	01	1B	33	21	DC	16	00	00	00	00	CF	FF	F1	FF
0000.104737750000000	FE	16	12	01	00	03	00	FF	BC	0E	12	01	01	1D	25	DA	05	00	41	CB	6E	44	AE	47	0F	BE
0000.111327250000000	55	55	26	02	BF	47	10	B7	BE	1C	CF	98	04	06	8C	00	00	00	03	00	F9	FF	F0	FF	8D	68
0000.112668333333333	55	55	2B	03	BE	C2	E3	00	80	08	00	80	08	00	80	08	00	2B	78	05	88	82	AB	FF	FF	FF
0000.113716083333333	FE	1A	4D	01	00	02	00	01	C0	1E	98	FF	01	FF	F0	E7	08	00	08	00	00	00	05	08	B7	02
0000.114885333333333	55	55	06	14	02	00	00	00																		
0000.131321750000000	55	55	26	02	C0	47	10	B7	BE	1C	CF	98	04	06	8C	00	00	00	F0	FF	06	00	06	00	8D	68
0000.132593416666667	55	55	18	00	BF	89	E3	00	80	08	00	80	08	00	80	08	00	2B	78	05	88	82	AB	FF	FF	FF
0000.133752500000000	FE	1A	4E	01	00	02	00	01	BF	1E	98	FF	01	FF	F0	E7	11	00	04	00	00	00	05	08	B7	02

4373 Hex /home/he/Daten/temp/SR24data/neu20/drone_yaw_180CCW_ch0.txt System time [UTC]: 2024-08-20 14:39:14

With the tool you can inspect single files or merge the files from two or four channels together to see what is going up and down.
One interesting point is that there are messages, Msg_ID=255 from/to cam (magic byte 0xFE) where another message format is nested in. Those nested messages with magic byte 0xBC are recorded in FlightLogs on ST16 as Sensor_xxxxx.bin. Sensor files from FlightLogs can be extracted with q500log2kml.

SR24 (RC) to Autopilot communication:

The basics are pretty clear. ST16 sends Channel data messages (Msg_type 0, 1 or 3) up to drone (Autopilot) and receives Telemetry messages (Msg_type 2). Bind message is Msg_type 4. All those messages are well documented and are enough to use STxx for other projects.

The tricky part is Msg_type 20. This is H480 specific. Some messages are clear, some not. Surprisingly I see no messages when I start calibration, also GPS switch looks strange. Here we need to dive deeper.

Autopilot to Gimbal/Camera/5GHz WiFi:

For feature "Dual Band Control Redundancy" Channel data will be sent via WiFi (and then camera UART). Backwards Telemetry messages are sent to Remote control. Both messages are pretty clear (Msg_ID=8), Telemetry over 5GHz is Msg_ID=2.

Currently I try to find out what the first 20 byte in Message "Attitude", Msg_ID 2 mean.

We can also exclude the Sensor messages (Msg_ID=255). In this messages is another message format embedded that is recorded in FlightLog files Sensor_xxxxx.bin. I think this is only for troubleshooting. Only Yuneec knows what there is in. I found only the System time and some position data (coordinates). This is for later development...

Communication rules found so far:

ST16 (Analyzer CH2) <-->	*Flight controller via SR24 (Analyzer CH3)*
ChannelData12 frequenty	
ChannelData12+Controller GPS	
	Telemetry
Commands via AdditionalData	Responses to Additional commands + RSdepth (maybe this message contains also depth from Sonar)

Flight controller serial (CH0) <-->	*Gimbal/Camera (Analyzer CH1)*
Heartbeat (1Hz broadcast) *	Heartbeat (1Hz) *
Attitude (frequently)	Attitude Status (1Hz) *
	Gimbal position (1Hz)
Telemetry to WiFi	

SensorData (frequently)	SensorData (similar to MAVlink V1 in some cases) MsgType 0 undef (1Hz) MISSION_REQUEST_INT (0,1Hz - means all 10sec) Controller GPS to WiFi Parameter request to Target ID 5 (SYS_SW_VERS)

* Those messages are always the same as I could see - can be taken to mimic H480 for other projects.

Heartbeat from Autopilot broadcast:	00 00 00 00 01
Heartbeat from camera:	00 00 7D 00 01
Attitude from gimbal:	01 00 01 maybe one is gimbal and the other camera
	06 00 01 see unknown SysID=6 that sends Telemetry_5GHz

Implemented RC lost procedure (Failsafe):

SR24 still sends Messages 0 and 3 but package counter begins to count from1 to FF (5s) and remains at FF. Messages have the values last sent by the RC although RC was switched off (forever I think, at least until I switches off the machine). FC still sends Message 3 (Telemetry) and message 20 (Additional data, action type 1 (till 9s), 3 and 8 from FC) for next 13s. Then FC stops sending anything.

```
11.5429178333 package counter starts to count. FC works as usual, sends message 3 and 20
13.5323969166 package counter reaches 100
16.6472205833 package counter reaches 255
24.5301494166 last message 2
24.3336494166 last message 20, msg 0 and 3 still coming from an not existing RC
29.1869264166 switched off all
```