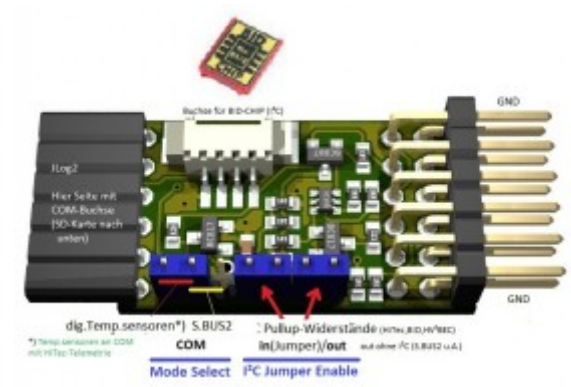
Anleitung JSend



Bitte lesen Sie als erstes diese Anleitung.

!!! Der JSend ist im Auslieferungszustand bereits auf die Anbindung JLog2-Futaba gejumpert !!!

Da das noch etwas Zeit benötigt, ist die heute publizierte Anbindung JLog2-Futaba nur als “Antifrust-Übergangslösung” zu verstehen. Sie funktioniert und bringt alle relevanten Sensordaten und Alarme mit, wird aber “final dann viel hübscher sein können”. – Außerdem benötigen Geräte, die nicht explizit für S.BUS II designed wurden, also auch das Fremdgerät JLog2, ein spezielles Stückchen Hardware, um sich an den S.BUS II anschließen zu können (Signale sind invertiert), siehe unten.

Der Early Bird beruht zum Einen darauf, dass zumindest der Sender T18MZ (die T-Box kenne ich nicht, wird aber soweit identisch sein) gegenwärtig vier Sensortypen anbietet, wovon “Temperatur” und “Drehzahl” sich sehr gut für das Übermitteln sämtlicher Daten von JLog2 verwenden lassen, einschließlich Möglichkeiten zur Alarmdefinition im Sender darauf. Zum Anderen ist es die herausragende Konfigurierbarkeit des Senders, die das in einer Weise ermöglicht, die relativ wenig an Mängeln über lässt, die man eben unter “Kompromiss” zu verbuchen hat. Im Wesentlichen ist dieser Kompromiss nur die automatische Verwendung der Maßeinheiten “°C” bzw. “rpm” auf einige JLog-Sensordaten, wo sie nicht passen. Dasselbe dann natürlich in der Sprachausgabe. – Aber damit sollte man entspannt leben können im Übergang, finde ich, schließlich kann dafür dem Warten auf JIVE und JLog2 an FASSTest® unmittelbar ein Ende gesetzt werden.

Einstellen des Senders:

Das S.BUS2-Protokoll (schreibt sich so einfacher ☺) beruht auf einem “Frame”, ein Teil (Zeitbereich) jedes Frames wird verwendet für das Senden von Sensordaten an den Empfänger, der es per FASSTest® an den Sender (T18) oder T-Box oder Wi-Fi-Rx übermittelt. Vier Frames bilden einen “Zyklus”, d.h., jeder Frame überträgt ein Viertel des Sensoradressraumes, – während Servodaten in jedem Frame vollständig übertragen werden. - Sensoren können in insgesamt 32 Time Slots (Zeitschlitzen) übertragen, 8 je Frame, ergo 32 je Zyklus (4 Frames). Der Slot 0 (Null) ist reserviert für Empfänger (bis zu 2, primary und secondary), hier werden Linkqualität, Empfängerspannung und externe Spannung auf den Bus gepackt, sodass sie ein Flightrecorder aufzeichnen könnte. (Die Überlegung ist, evtl. diese Daten mit in’s Log von JLog2 zu nehmen.). Die 31 verbleibenden Slots können den Sensoren auf dem S.BUS2 zugewiesen werden. - Hat ein Sensor mehr als ein Datum zu liefern, ist ergo ein Multi-Sensor, so wie die Futaba-Sensoren “Vario” und “GPS”, so wie JLog2, Unilog2, GPS-Logger etc. pp., dann hat er dafür fortlaufende Slots zu verwenden. Die erste Klausel ist aber, dass es nur 8 Slots pro Frame gibt. Ergo muss die benötigte Anzahl Slots eines Multisensors, abhängig von der Startadresse (der Nummer des ersten Slots) noch in einen Frame von 8 Slots passen, sonst kann er nicht in diesem Frame platziert werden, muss in einen anderen Frame, der genügend fortlaufende Slots frei hat. - Das “Vario” von Futaba benötigt 3 Slots, “GPS” 8 Slots, während der “Temperatursensor” nur einen Slot belegen will. JLog2 will z.Z. 12 Slots, benötigt also einen kompletten Frame (8 Slots) plus 4 fortlaufend freie Slots in einem anderen Frame.

Frame 0: Slot 1…7 (0 durch Rx besetzt), Frame 1: Slot 8…15, Frame 2: Slot 16…23, Frame 3: Slot 24…31

JLog2 belegt fest (in der Firmware z.Z. eingebrannt) Slot 16…23 und Slot 24…27, mit der Option, demnächst auch Slot 28…31 zu beanspruchen (für Daten von einem Staurohr (Speed) und vom HV²BEC von Linus), also die kompletten Frames 2 und 3. – Zu beachten wäre, dass Telemetriedisplays in Seiten von je 9 zur Anzeige gebracht werden (T18MZ), und zwar in der Reihenfolge der Slot-Adressen 0…31, sofern der jeweilige Slot im Sender für einen Sensor definiert ist, nicht inaktiv ist. – Registrierte Sensoren, dem Sender funktionell bekannt, besetzen durch ihre Aktivierung im Setup des Senders automatisch (oder ebenso automatisch durch den angeschlossenen Sensor) und fortlaufend die benötigten Slots, “Vario” 3, “GPS” 8. Beim Automatismus macht der Sender das Slot-Management von selbst, versucht also z.B., “GPS” 8 fortlaufende Slots zuzuweisen. Will man manuell “GPS” aktivieren, wird es einem erst gar nicht angeboten als Sensortyp, wenn inklusive des manuell gewählten Start-Slots gerade nicht fortlaufend 8 Slots undefiniert sind, – ähnlich bzgl. “Vario”, nur, dass es sich hier um 3 Slots handelt.

Nachdem man das verinnerlicht hat, geht’s an’s Definieren von Slots für JLog2, also der Slots 16…23 und 24…27, wobei man auch gleich Slots 28…31 für JLog2 definieren kann wegen der fortlaufenden Anzeige, 28…31 werden alsbald durch spezielle Firmwareversionen von JLog2 beansprucht werden, im Augenblick, mit der Standard-Firmware, aber nicht. Wir verwenden die Sensortypen “Temperatur” und “Drehzahl” für alle 12 (bzw. dann 16) Daten von JLog2:

Slot 16: DZ “Ibec (A) *10″ Slot 17: TEMP “Ubat (V)” Slot 18: DZ “Imot (A)” Slot 19: TEMP “throttle (%)” Slot 20: TEMP “PWM (%)” Slot 21: DZ “RPM[uni]“ Slot 22: TEMP “tFET” Slot 23: DZ “mAh” – Slot 24: TEMP “ext: t1″ Slot 25: TEMP “ext: t2″ Slot 26: TEMP “ext: t3″ Slot 27: DZ “ext: RPM”

Slot 28: DZ “ext: SPEED(km/h)” Slot 29: TEMP “ext: Ubec (V)” Slot 30: DZ “ext: Ibec(A) *10″ Slot 31: TEMP “ext: TEMP hvBEC”

Ibec wird *10 angezeigt, man muss sich eine Kommastelle von rechts denken. Das “[uni]” in “RPM[uni]” bedeutet, man kann Motordrehzahl oder Rotordrehzahl anzeigen lassen (JLog2 liefert hier RPMmot), je nachdem, was man als Ratio (Untersetzung) hinter diesem Wert im Sender wählt, z.B. 1:1 für Motor oder 1:12.82, wenn Ritzel==17 und HZR==218. — Es werden nur die ersten 3 der Werte von bis zu 5 digitalen Temperatursensoren in die Telemetrie geschickt, – Die Maßeinheit “°C” bzw. “rpm” wird vom Sender automatisch eingeblendet, muss man ignorieren, wenn nicht zutreffend, dafür steht ja die Einheit im Namen des Meßwertes. – Meßwertnamen (Sensornamen) können max. 16 Zeichen lang sein. Beim Sensortyp “Drehzahl” wählen wir “Magnet.” als Subtyp, leider wird der Subtyp dann auch als String “Magnet.” an den von uns gewählten Sensornamen vom Sender angehängt. Man kann den String, so weit es geht, nach rechts aus der Namenanzeige

herausschieben, indem man den jeweiligen Namen eines Sensors vom Typ “Drehzahl” mit Leerzeichen auffüllt, bis die max. 16 Zeichen erreicht sind.

*Auf besonderen Wunsch machte ich zwischenzeitlich auch eine Firmware für JLog2, die in Slot17 Ubat*10 auf einen Sensor Typ “RPM” statt Ubat auf “Temp” liefert. Dadurch kann mit einem gedachten Komma auf 0,1V genau angezeigt werden. (Die Anzeige einer Temperatur hat nur den Wertebereich -100..155, ohne Kommastelle, während “RPM” von 0.. “sehr viel” läuft, ebenfalls ohne Kommastelle, natürlich.)*

Setup im Ablauf:

Grunddisplay –> Basis –> Seite 2 –> Sensor: Hier definieren wir die Zeitschlitze 16…27 bzw. gleich 16…31, d.h., zunächst nur den Sensortyp “Temperatur” oder “Drehzahl nach obiger Vorgabe.

Dann: Grunddisplay –> Basis –> Seite 2 –> Sensorname: Das Umdefinieren der Defaultnamen “Temperatur” bzw. “Drehzahl” in die obigen Namen erfolgt nach Sensortypgruppen, “Temperatur” und “Drehzahl”.

Nun wird das Ratio für “RPM[uni]” und Alarme auf Sensoren eingestellt, sofern sinnvoll und gewünscht: Grunddisplay –> Basis –> Seite 2 –> Telemetrie (oder man tippt auf den Bereich mit der Empfängerspannung im Grunddisplay): Dann tippt man auf einen Sensor (sein Display) und landet im betreffenden spezifischen Setup. – Alarm: “INA” bedeutet inaktiv (administrativ abgeschaltet), nicht verwirren lassen, “AUS” bedeutet nicht “ausgeschaltet”, es bedeutet, dass gerade keine Alarmbedingung vorliegt. Man aktiviert nun einen Alarm auf Überschreiten eines einzustellenden Schwellwertes oder/und Unterschreiten eines zweiten Schwellwertes. - Bei Alarm piept der Sender immer, das lässt sich offenbar nicht abstellen. Man kann nun einen von 5 Vibrationsalarmen on top konfigurieren und außerdem bei Bedarf die Sprachausgabe von “INA” auf “AKT” stellen, aktiviert.

Das Verhalten der T18MZ bei Alarm ist wie folgt (wobei ich da ein paar Verbesserungsvorschläge hätte):

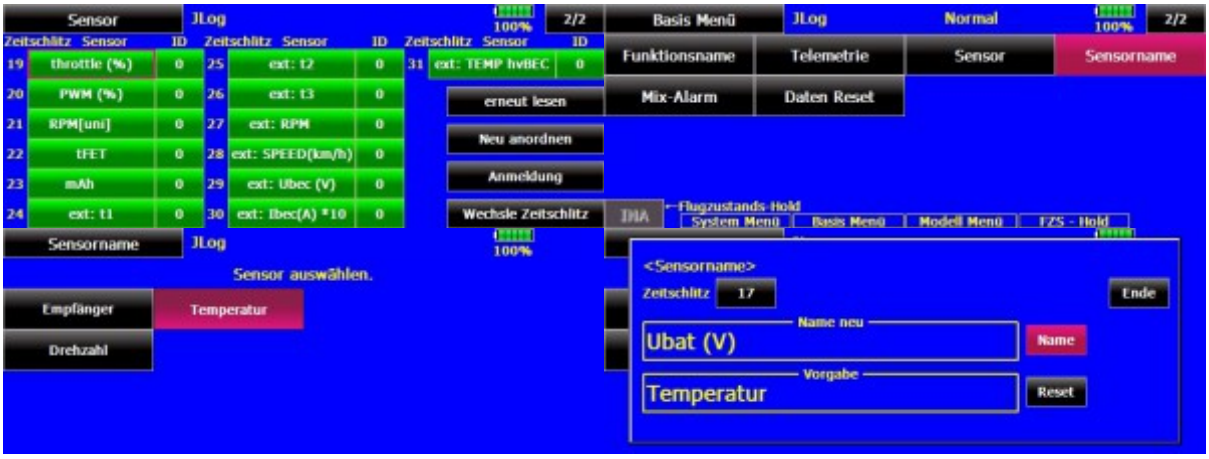
Befindet man sich nicht in der Telemetrieanzeige, popt diese auch nicht hoch, wenn ein Sensor einen Alarm meldet, nur das Akustische und der Vibrationsalarm melden sich ergo. In der Telemetrieanzeige wird das betreffende Sensordisplay invertiert. – Vibration und akustischer Alarm werden bei aktivem Alarm durch den Sensor bestimmt, der in der Anzeigereihenfolge am weitesten vorn steht. Sensoren in der Reihenfolge dahinter signalisieren in diesem Augenblick ihren Alarm nur visuell durch Invertieren. Wenn man bei Alarm also nicht visuell checkt, sollte man aufpassen, wenn man einen Alarm eines Sensorwertes vor einem anderen bewusst ignoriert, man könnte den Alarm eines nachfolgenden Sensors verpassen, mit evtl. bösen Folgen! – (Logischerweise werden bei Sprachalarmen bzw. bei ständiger Werteansage hier oft die falschen Bezeichnungen bzw. Einheiten gesprochen, also “Temperatur”/”°C” oder “Drehzahl”/”Umdrehungen”. - Es gibt offenbar keinen Alarm auf Empfangsausfall, man muss einen Alarm auf die Empfängerspannung gesetzt haben, die geht auf Null, wenn der Rückkanal des Rx nicht mehr vom Sender gesehen wird. Witzigerweise behalten die anderen Sensoren den letzten gesehenen Wert im Display, wenn von denen nichts mehr empfangen wird.) – Vorteil mit Futaba: Im Alarmfalle wird auch der aktuelle Wert angesagt.

Das ist die neue Standard-Firmware 3.2.2 H4F-32 für Jlog2, die S.BUS2 unterstützt, – für das Konfigurieren dessen wird wiederum der passend erweiterte JLC 4.4.2.5 benötigt.

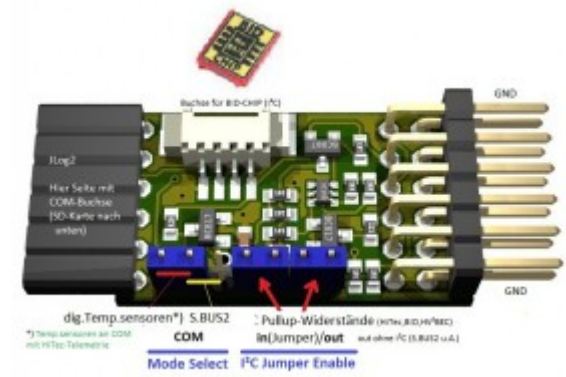
Spezialfirmwareversionen für JLog2 mit S.BUS2-Unterstützung werden folgen, z.B. eine neue “T”-Version (mit JLog-eigenem Speed-Sensor (Prandtl Probe)) und eine für den HV²BEC.

Im Futaba-Forum meinte jemand mit Bezug auf den betreffenden Thread im JLog-Forum, das wäre zumindest ein Lichtblick für Kontronik-Jünger. Bitte nicht vergessen, dass JLog2 auch ohne JIVE betriebsfähig ist, er bringt dann die Daten eigener Sensoren in’s Log und z.T. auch in die Telemetrie. Das ist teilweise die Frage einer passenden speziellen Firmware, die auf den Logger zu flashen ist, weitere spezielle Versionen zum überbrückenden Heilen besonderer Bedürfnisse von Futaba-Anwendern wären nicht generell auszuschließen. – Allerdings wäre zum Beispiel ein extra

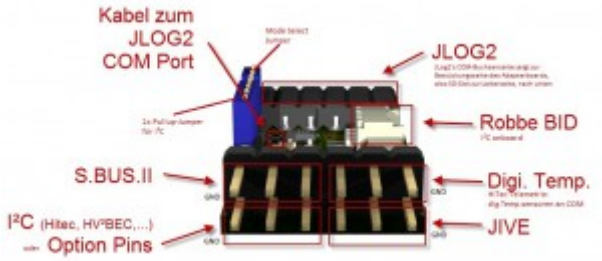
Stromsensor eher eine Domäne für Unilog2, der demnächst vermutlich auch den S.BUS2 bedienen können wird.



Weil der S.BUS2 (genauso wie S.BUS) invertierte Signale verwendet, außerdem tri-state sein muss, und weil das nicht per Software im Microcontroller machbar ist, und eine dritte Signalleitung zum Steuern des Z-State (Bus-Anschluss floating) nicht z.V. steht, blieb mir nichts anderes übrig, als eine Adapterschaltung zu entwickeln, die den Z-State automatisch einnehmen kann. Linus war dann so freundlich, sich anzubieten, so ein Board als fertiges Interface in Kleinserie zu fertigen: Interface-Adapter für JLog2–S.BUS2 und JLog2 an I²C-Devices (HiTEC-Telemetrie, HV²BEC-Setup/Logging/Telemetrie, später evtl. Spektrum- und/oder JR-Telemetrie, falls das wirklich auch I²C-basierend sein sollte.), digitale Temperatursensoren ersatzweise am COM-Interface für HiTec-Telemetrie (weil die zwei Optionalpins bei I²C durch den Bus besetzt sind), BID-Chip (auch I²C). Damit wird also von Anwendern entsprechenden Equipments (Futaba-Telemetrie, HiTEC-Telemetrie, BID-Chip, HV²BEC) die Last des Bastelnmüssens genommen. (Ich habe daher den Schaltplan im Artikel oben wieder herausgenommen.) Alle anderen Anwendungen mit JLog-eigenen Sensoren und Futaba-Telemetrie sind wie gehabt möglich, da das Optionalinterface von JLog2 auf der Platine durchgeschleift wird. – Danke,



Linus!!!



Der Logger wird auf der rechten Seite direkt auf das Board gesteckt, die Tiefe der 6-poligen Buchse geht also von der Gesamtlänge ab. Links dann 1) Patchkabel zum JIVE, 2) Kabel zum HiTec-Empfänger bzw. HV²BEC bzw. durchgeschleifte JLog-Optionspins (JLog-eigene Sensoren), 3) digitale Temperatursensoren (Dallas-1-Wire-Bus) bei I²C-Nutzung (HiTEC etc.), 4) Robbe/Futaba S.BUS2. Die Buchse im Vordergrund ist zum Anstecken eines Robbe BID-Chips. Auf der Platine 3 Lötpads für das Kabel zur COM des Loggers (Molex-Stecker am Ende). Drei Jumper: 1 und 2 zum Aktivieren/Deaktivieren der Pullups für I²C (HiTEC, HV²BEC, evtl. auch BID-Chip), 3 zum Umstecken zwischen S.BUS2-Konfiguration und Nutzung der COM für digitale Temperatursensoren mit I²C (HiTEC, ..).

Die Platine ist etwas länger als JLog2. – Für alle anderen Telemetrien und Anwendungen des Loggers erbringt das Board keinen Vorteil.

———-

Ja, und auch mal ein Danke an Robbe! Immerhin hat man einen Fremdsensor unterstützt, bevor man selbst Sensoren auf dem Markt hat, was, strategisch gesehen, schon ein deutlich positives Signal darstellt, oder? Will sagen, das sollte die gefrusteten Wartenden schon optimistisch stimmen. 😃

P.S. (15.7.2012) Die Lieferbarkeit der relevanten Artikel, wie seit geraumer Zeit in Robbe's Webshop gelistet, scheint nun unmittelbar bevor zu stehen. Robbe wird der JLog-Entwicklung auch eine Telemetriebox z.V. stellen.

_____

Tja…, nun wäre Gelegenheit, das Upgrade auf JETI v1.1 (EX) einzuschieben. Das passt ja noch in die Telemetrieübersicht, was aber, wenn wider Erwarten hinreichende Informationen zur Telemetrie von Spektrum oder JR eintrudeln sollten? Beim besten Willen, das darf nicht passieren, kein Platz mehr im Bild. .

_____

Mal zu den Begrifflichkeiten, das Kind muss ja einen Namen haben, aber es hat drei:

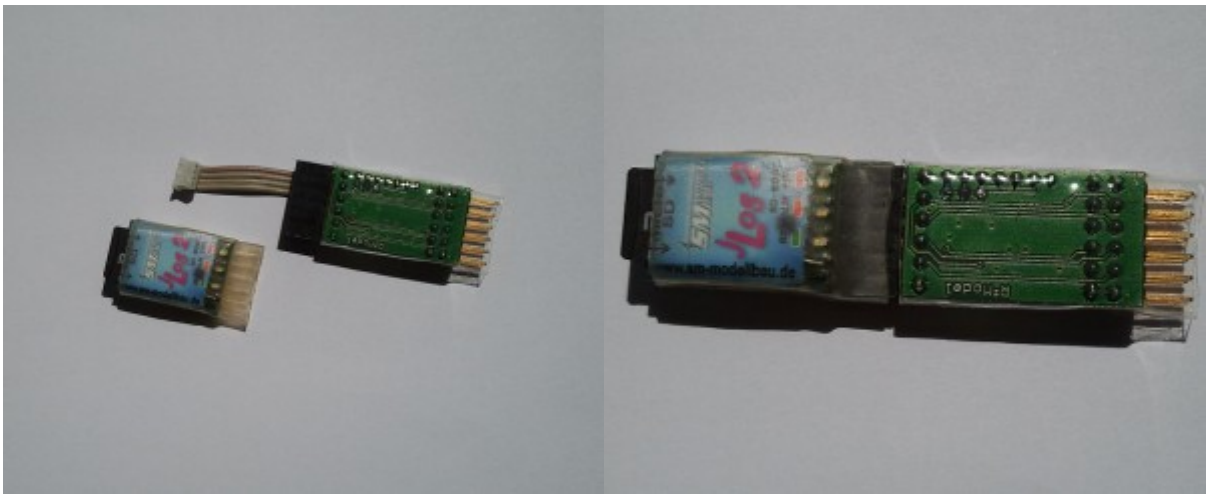
Man könnte nun “Futaba-Telemetrie” sagen, es gibt ja nur eine, bzw. ist sie am Kommen. Einfach “Futaba” oder “Robbe/Futaba”, und fertig.

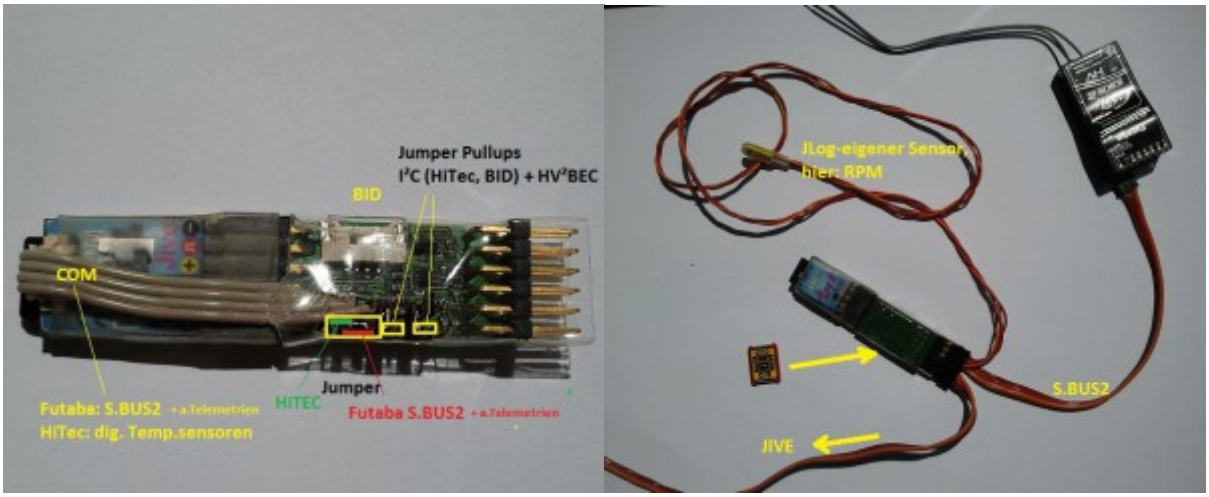
FASSTest® steht für das Protokoll, was auf dem HF-Wege gesprochen wird, Sender–>Empfänger, Empfänger–>Sender|T-Box|WiFi-Rx, wobei letzterer Weg für die Telemetrie entscheidend ist.

S.BUS2 ist der Bus, auf dem Sensoren an den Empfänger ihre Daten senden, – aber der S.BUS2 kann, wie gesagt, auch für sehr viel mehr zuständig sein. Er beschickt z.B. die Servos, wenn dies nicht via *S.Bus* oder herkömmlich über Servoimpulse an Rx-Einzelkanälen erfolgt.

Indem ich sage, “mein Sensor spricht S.BUS2″, ist es dasselbe wie “er ist der Futaba-Telemetrie fähig”, damit ist definiert, dass diese Daten per FASSTest® nach unten zum Sender gelangen, alternativ zu einer T-Box oder einem WiFi-Rx. Keine Sensordaten via S.BUS2 ohne FASSTest®, kein FASSTest® ohne S.Bus2, sie benötigen einander. Nur ein Empfänger, als Ambassador des Senders auf dem S.BUS2, kann als Sensor den FASSTest® verwenden, ohne einen S.Bus2 bemühen zu müssen. Logisch, ER ist ja derjenige, der das FASSTest® “nach unten” spricht.

.





T18MZ, T-Box [, WiFi-Rx]

Nachdem die T-Box nun käuflich ist, stellt sich offenbar heraus (Ich habe noch keine T-Box.), dass diese nicht so mit JLog2 funktioniert wie es der Logger mit der T18MZ tut. Daher erscheint es mir angebracht, noch mal etwas zu den Prinzipien der Telemetrie von Robbe/Futaba zu sagen:

Der S.BUS2 ist das Interface für Telemetriesensoren. Der Bus ist universell ausgelegt, ja, Sensordaten über ihn zum Empfänger zu schicken, ist nur eine seiner Funktionen. Allerdings trägt die Definition des Busprotokolls rein weg gar nichts zu der Frage bei, wie übertragene Sensordaten am Ende dargestellt und verwertet werden. “Am Ende” bedeutet im Terminal, Terminal ist ein Sender (i.Augenblick nur T18MZ), die T-Box oder der WiFi-Rx (noch nicht erhältlich).

Schauen uns wir mal die verschiedenen Prinzipien um die Darstellung von Telemetriedaten an:

Die flexibelste Form, das Terminal muss überhaupt nichts von darzustellenden Daten wissen, ist, wenn der Sensor selbst und direkt das komplette Display gestaltet. Das macht JETI so in der Basisform v1.0, die auch in EX enthalten ist, HoTT macht es so im Textmode. HoTT hat den Vorteil, dass es sich um eine Punkt-zu-Multipunkt-Verbindung handelt (adressierte Displays), während JETI Punkt-zu-Punkt ist, einen Expander benötigt für mehrere Sensoren als Quelle. - Datenauswertung im Terminal (z.B. für Alarme) kann nicht erfolgen, weil die erfordern würde, dass der Aufbau jedes Displays (jeder Seite) bekannt ist, also die Datenpositionen und die jeweilige Datendarstellung. Volker macht es aber trotzdem mit seinem “VSpeak” für JETI. Alarme werden ergo von den Sensoren gebildet, das Terminal muss als Alarmgeber vorbereitet sein, also dedizierte Alarme anbieten, die die Sensoren triggern.

Bei Multiplex (M-Link, MPX Sensor Bus – MSB) gibt es eine Reihe definierter Datenklassen, ein Sensor sendet seine Daten auf eine abgefragte Adresse unter Angabe der Datenklasse, inklusive eines Alarms, wenn der in einer Datenklasse möglich ist. Das Terminal (der Sender) weiß also anhand der Datenklasse, wie ein empfangenes Datum darzustellen ist. Das System ist so flexibel wie der Umfang möglicher Datenklassen hoch ist und die Datendarstellung in den Klassen für einen Sensor adäquat. Leider hapert es da noch ein bisschen beim MSB, zu wenig Datenklassen, die teilweise nicht sehr praxisnah sind, zu wenig mögliche Adressen (16). Das Prinzip an sich ist aber mMn genau das richtige.

HoTT im Binärmodus definiert einen Datensatz pro Sensortyp, der Datensatz ist gemäß Busprotokoll durch den Sensor via HoTT-Bus zu befüllen. Je Datensatz (Sensortyp) ist ein Display fest in der Firmware des Terminals (Sender) eingebrannt. Insofern ist das hochgradig inflexibel, fix eben, aber die Darstellung der Daten in einem Display ist bekannt, wenn der Sensor diese auf den Bus gibt. Der Nachteil ist nur, dass Sensoren, die nicht zum System gehören, eigentlich ein eigenes Display (Datenblock + Darstellung) benötigen, was ein Aufbohren der Firmware des Terminals erfordert, also durch Graupner. Fremdsensoren “vergewaltigen” daher oft bestehende Displays, momentan

GAM/EAM/VARIO/GPS, um irgendwie darstellungsrichtig ihre Daten an den Mann bringen zu können.
- Das Terminal ist in der Lage, die Daten auszuwerten, es kennt deren Position und Darstellungsform. Genutzt wird das für Sprachausgaben, Alarme werden aber trotzdem durch die Sensoren gebildet, das Terminal muss sie nur ausgeben können, vor allem in Form von Sprachdateien (Audio).

Bei HiTec ist das ähnlich, ebenso bei Spektrum und vermutlich auch JR, nur, dass es nur ein Display mit spezifischen Datenfeldern gibt und zumindest bei HiTec sogar das Data Processing teilweise im Terminal (Sender) implementiert ist, z.B. die Linearisierung der Strommesswerte von einem Hall-Core. Ein Fremdsensor muss dann ein entsprechendes Datum vorverzerrt an den Sender geben, damit der nach der Linearisierung auch den gewollten Wert darstellt.

JETI Ex setzt auf das Basisprotokoll v1.0 (Textterminal) erweiterte sog. “Hidden Messages” mit v1.1 oben drauf. Der Sensor kann wie bei Multiplex Datenklassen für seine Ausgaben beanspruchen, bekommt je Datum automatisch ein Subdisplay, und er kann sogar die Maßeinheit und Beschriftung frei bestimmen. Die Daten sind binär, das Terminal kann sie also interpretieren, hier nur genutzt zur Sprachausgabe, Alarmbildung erfolgt nach wie vor im Sensor.

Alarme ganz allgemein: Differenzierte Alarme sind das, was das Salz in der Telemetriesuppe ausmacht. Einige Systeme glänzen aber immer noch mit EINEM Generalalarm, z.B. Multiplex und HiTec, wenn man nicht den externen Voice-Modul hat. JETI vor der JETIbox Profi und JETI-eigenen Sendern bot ein bestimmtes Morsezeichen pro akustischem Alarm. Nun, Sprachalarme sind wohl angesagt. Teilweise gibt es auch Vibrationsalarm (on top), bei Robbe/Futaba sogar verschiedene Sequenzen (wählbar je Alarm). Die Crux ist nur, dass, egal, ob der Sensor oder das Terminal den Alarm triggert, passende Audiofiles durch das Terminal vorgehalten und zugeordnet werden müssen. Bei JETI kann man wohl selbst relativ leicht solche Files einbringen, Robbe/Futaba wird demnächst hier volle Flexibilität durch synthetische Sprachausgaben erlangen.
Wo der Alarm gebildet wird, ist zunächst mal egal, ob im Sensor oder im Terminal. Im Terminal wird es von der Bedienung her einfacher sein, denn, macht man es im Sensor, sollte man den auch vom Terminal aus konfigurieren können. Bei JETI ist das so, das Terminal ist eh ein “dummes”, bei HoTT erfolgt das im Textmode. – Allerdings kann es u.U. von großem Vorteil sein, wenn der Sensor den Alarm bildet. Der Sensor ist quasi eine State Machine, er weiß anhand des Betriebszustandes, wann ein Alarm z.B. zu deaktivieren ist, auch wenn der getriggerte Wert noch unter Alarmbedingung steht.

Nun zurück zur Telemetrie von Robbe/Futaba: Dadurch, dass die Dateninterpretation rein durch die Firmware des Terminals geschieht, das Busprotokoll hier nichts beiträgt, ist es erforderlich, dass JEDER Sensor in den Terminals zu implementieren ist! Es gibt dann dieses Akkreditieren von Sensoren am Terminal (einmal direkt anzuschließen), das vereinfacht dem Anwender die Zuordnung der 31 nutzbaren Timeslots zu den Daten der verschiedenen Sensoren. Damit das funktionieren kann, muss das Terminal aber den Sensor kennen, seine unique ID, die Robbe/Futaba vergibt, seine Struktur (Daten je Slot nach Darstellung und Reihenfolge, wenn mehrere Slots beansprucht werden).

Die gegenwärtige JLog-Lösung mit der T18MZ beruht darauf, dass der Sender alle bekannten Sensoren dem Anwender zur freien Slot-Zuordnung im Setup anbietet, dieses Akkreditieren ist eine Altnernative, die automatische Slot-Zuordnung ermöglicht. Die Vorabvariante mit JLog2 beruht ergo auf manueller Zuordnung, wobei die Firmware des Loggers Slot-Adressen fix besetzt. (Durch Akkreditieren teilt das Terminal dem Sensor mit, welche Slot-Adressen er verwenden soll.)
Der Punkt mit der gegenwärtigen Firmware der T-Box scheint nun aber zu sein, dass diese im Gegensatz zur T18MZ eben NICHT bekannte Sensoren zur manuellen Slot-Zuweisung anbietet, sie unterstützt nur das Akkreditieren, – und das kann mit JLog2 noch nicht gehen, weil er noch nicht in die Terminals implementiert worden ist. Und bingo.., war’s erst mal ein Satz mit X… Ziemlich unnötiger Frust, das… Nun heißt es wieder Warten auf Robbe/Futaba, es sei denn, Robbe macht die T-Box vorher wenigstens funktionell identisch zur T18MZ.

Ganz allgemein gesprochen: Was der S.BUS2 an Universalität einbringt, verliert das System im Telemetriebereich wieder durch unflexible Datendarstellung bzw. -interpretation. Die Time Slots des Busses für Sensordaten lassen in der Definition des Protokolls leider alles offen, was Datentyp und Wertebereich betrifft. Im Ergebnis ist das Ganze sehr unfreundlich zu Geräten von Drittanbietern, so bemüht auch Robbe ist, 3rdParties integrativ zu unterstützen. Die Neuerungsrate gegenüber dem

Markt wird dadurch potentiell einer Bremse unterworfen, jeweils notwendige Nachentwicklungen bei Robbe/Futaba sind das installierte Nadelöhr.

Altgeräteentsorgung

Elektronische Geräte dürfen nicht einfach in eine übliche Mülltonne geworfen werden. Der Artikel ist daher mit dem nebenstehenden Symbol gekennzeichnet. Dieses Symbol bedeutet, dass elektrische und elektronische Geräte am Ende ihrer Nutzungsdauer, vom Hausmüll getrennt, entsorgt werden müssen. Entsorgen Sie das Ladegerät bei Ihrer örtlichen kommunalen Sammelstelle oder Recycling-Zentrum. Dies gilt für alle Länder der Europäischen Union sowie anderen Europäischen Ländern mit separatem Sammelsystem.

Konformitätserklärung

CE

Hiermit erklärt MHM-Modellbau KG,
dass sich das Produkt in Übereinstimmung mit den grundlegenden Anforderungen und den übrigen einschlägigen Bestimmungen der Richtlinie 2014/30/EU und 2011/65/EU befindet.

MHM Modellbau®

MHM-Modellbau KG

Neudorfer Str. 281 F
09474 Crottendorf
WEEE-Reg.-Nr. DE 41692360
www.mhm-modellbau.de