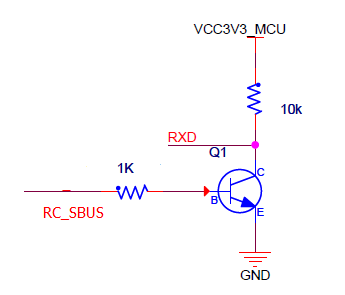
1. **Theory of Operation.**

The SBUS Protocol is a serial protocol that was developed by Futaba for remote control applications. It’s used to access up to 20 signals, including 16 channels, 2 digital channels, and one failsafe plus one framelost flags.

It’s the same as UART protocol, the only difference is the voltage levels are inverted. Therefore, in order to read the proper UART signal sent by SBUS, we use a signal inverter circuit:

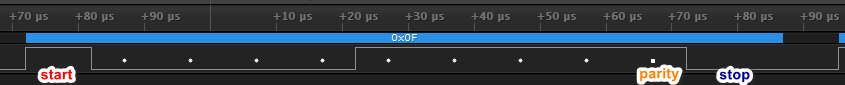


For reading SBUS message, a serial port has to be configured as follows:

* 100 000 Baud rate

8E2 configuration, i.e.:

* 1 start bit
* 8 Data bits
* 1 Even parity bit
* 2 Stop bits

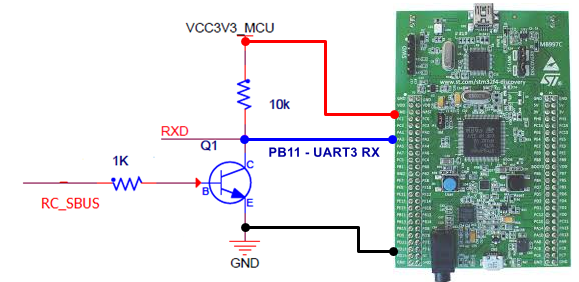


*Zoom in of SBUS message’s first byte*

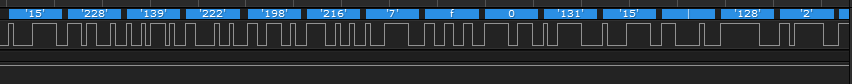
SBUS messages are sent periodically every *6.35 [ms].* A single SBUS message is *25 [bytes]* long. Therefore, with the configuration described above, it takes slightly more than 3 [ms] ( *25\*(1+8+1+2)[bits]/100000[bits/s]* ) for a SBUS message to be transmitted, consists of the following bytes:

* 1 Header byte 00001111b (0x0F).
* 22 bytes -> 16 [channels]\* 11 [bits].
* 1 Byte with two digital channels (channel 17 and 18) and "frame lost" and "failsafe" flags.
* 1 Footer byte 00000000b (0x00).

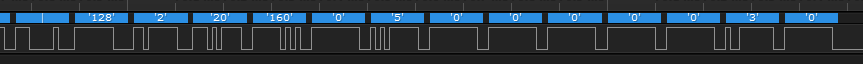
1. **Decoding sbus channels.**



*STM32F4 Discovery and SBUS circuit*



*Capture of several UART bytes from SBUS start message*



*SBUS end message*

A single SBUS channel is 11-bit long, and the sbus channel signal bytes are [byte1:byte22].

Below is the relationship between SBUS signal bytes and channel signals,

With the notation*:* cx.y *means channel x bit y.*

SBUS byte1: [c1.7 c1.6 c1.5 c1.4 c1.3 c1.2 c1.1 c1.0]

SBUS byte2: [c2.4 c2.3 c2.2 c2.1 c2.0 c1.10 c1.9 c1.8]

SBUS byte3: [c3.1 c3.0 c2.10 c2.9 c2.8 c2.7 c2.6 c2.5]

SBUS byte4: [c3.9 c3.8 c3.7 c3.6 c3.5 c3.4 c3.3 c3.2]

...

SBUS byte21: [c16.2 c16.1 c16.0 c15.10 c15.9 c15.8 c15.7 c15.6]

SBUS byte22: [c16.10 c16.9 c16.8 c16.7 c16.6 c16.5 c16.4 c16.3]

Thus, to get signals channel 1 and 2, we use the following pseudocode:

channel\_1 = byte2<<8 + byte1 = 0x[c1.10 c1.9 c1.8 c1.7 c1.6 c1.5 c1.4 c1.3 c1.2 c1.1 c1.0];

channel\_2 = (byte2>>3 | byte3<<5)\*0x7FF = 0x[c2.10 c2.9 c2.8 c2.7 c2.6 c2.5 c2.4 c2.3 c2.2 c2.1 c2.0];

**References**

1. <https://github.com/uzh-rpg/rpg_quadrotor_control/wiki/SBUS-Protocol?fbclid=IwAR1qP5Yy83gAx3_xcv6XjBVWboljo6V5k8zG_ekSbxFHFzUlQq8bN2AV70U>
2. <https://github.com/bolderflight/SBUS/blob/master/src/SBUS.cpp>
3. <https://os.mbed.com/users/Digixx/notebook/futaba-s-bus-controlled-by-mbed/>
4. <https://github.com/PX4/Firmware/blob/master/src/drivers/linux_sbus/linux_sbus.cpp>
5. <https://dev.px4.io/v1.9.0/en/tutorials/linux_sbus.html>