Datalogger Hardware Revision 1.2

Change Log

Pending

* Move simple components to 0805 surface mount, and this may become a double-sided design
*   Hand-route design for maximal prettiness, perhaps also add a ground plane / power plane for better voltage regulator heat dissipation
* Eliminate one supercapacitor
* Add diode onto 12v line to prevent leaking current onto the CANbus
* Fix programming header arrow, add shroud indication
* Tie MCP23017 reset to board reset
* Add RGB LED indicators, new proposed system:
  + For status lights:
    - Red (steady): non-operational, error state
    - Green (steady): operating normally
    - Yellow (blinking, on top of steady light): warning state (as in "SD card almost full)
    - Blue (steady): waiting for user state (as in "remove SD card" or "insert SD card")
  + For activity lights:
    - Red (steady): no communications / communications error
    - Green (blinking): received data / reading data
    - Blue (blinking): transmitting data / writing data
    - Cyan (blinking, combination of green and blue): both receiving and transmitting data
  + LEDs will be:
    - Activity (RGB)
    - CAN (RGB)
    - SD (RGB)
    - Fault (just red, directly tied into microcontroller)
* Make buttons longer
* Remove propeller and associated chips
* Move FT232 lights under the USB header, make them 0805 surface mount
  + Reasoning: this is used during debugging only so it would be unnecessary to give them full-size LEDs along with the other, more driver-relevant indicators.
* Add USB connection light to the FT232 chip
* Add bluetooth serial chip with relevant lights
  + Reasoning: Too often, we have the datalogger connected to the PC for debugging, and the wire is a real pain when the board is in the car.
  + Additionally, bluetooth chips aren't terribly expensive anymore. A Class I RN-41 costs $25, and the shorter range RN-42 is $15.
* Add either switch or diode circuit to not have both serial chips interfere with each other
* Marker-able area for board information like Pb-free and serial number
* Better switch labeling

Open Questions:

* Should we stick with the current PIC chip or go for a higher-end chip? The current one gives the necessary data rate, but there is a concern about overflowing on start up since the SD card takes a while (about a second) to initialize, and during that time there may be too many received messages and the buffer may overflow.
  + Note that the higher end PIC chips (32-bit MIPS processor) are 64-TQFP, which may be a reason to avoid them since SMT chips are not socketable and we probably won't have SMT rework equipment for that many pins in Australia
  + In general, most higher-end chips are ARM based and come in 100+-pin TQFP or BGA (!!!) packages
  + These chips would allow some complexity to be removed (no IO expander, less complicated buffering code since all memory is DMA) and may allow extra features (like Ethernet user interface / bootloader or receiving all messages on startup)
  + A dual-chip design with both the current PIC and a higher-end chip is likely not feasible. The reason this worked with the propeller was since both chips had remappable peripherals which allowed a relatively clean design, while higher-end chips have peripherals hard wired to pins

Implemented