DESCRIPTION

Unmanned vehicles have common needs: energy to think and move around, sensors to interact with their environment, communication skills and servo control. It really doesn't matter if the vehicle is going to fly, float, dive or move over hard surfaces. In most cases all of the above requirements are present in one form or another.

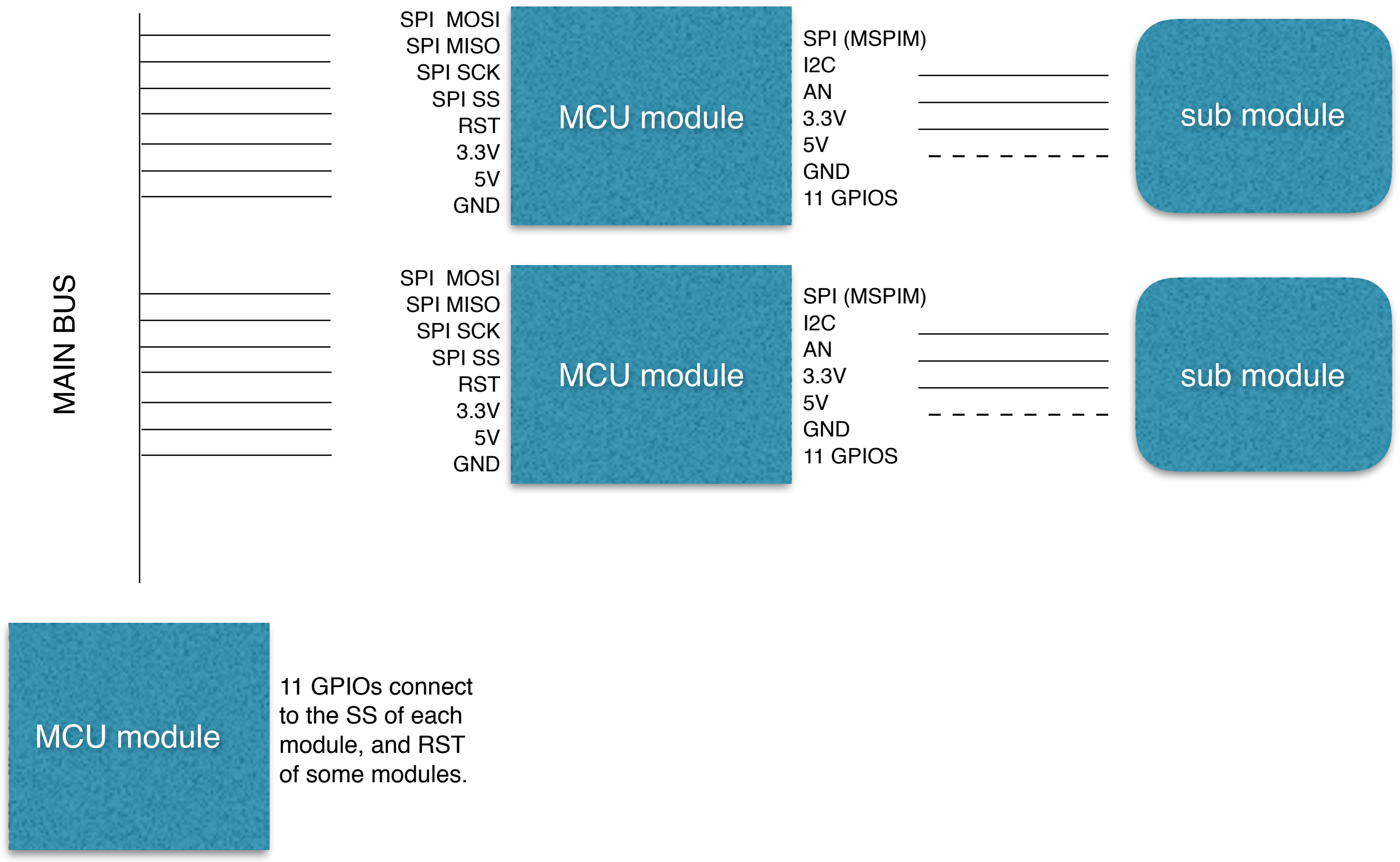
I've been working on this project for some time. It is a modular flight controller. It was made modular so I could work on one module without worrying about breaking down the rest of the system. I wanted to be able to upgrade a module with a new sensor, component or capability as much as I wanted the whole system to configure itself every time a new module was plugged.

A desired sub-product of this design should be the ability to handle redundancy. Sensor, pilot and engine redundancy. A die hard system.

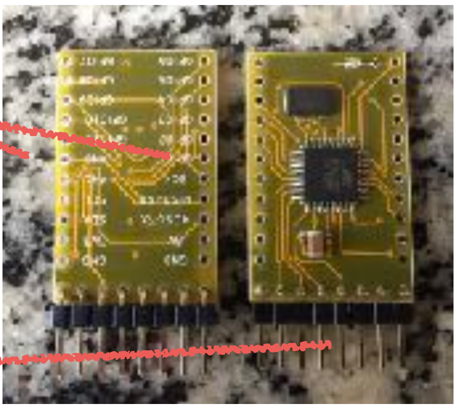
I decided to move this project to open source and open hardware. I hope to attract your interest and bring you to this new development community.

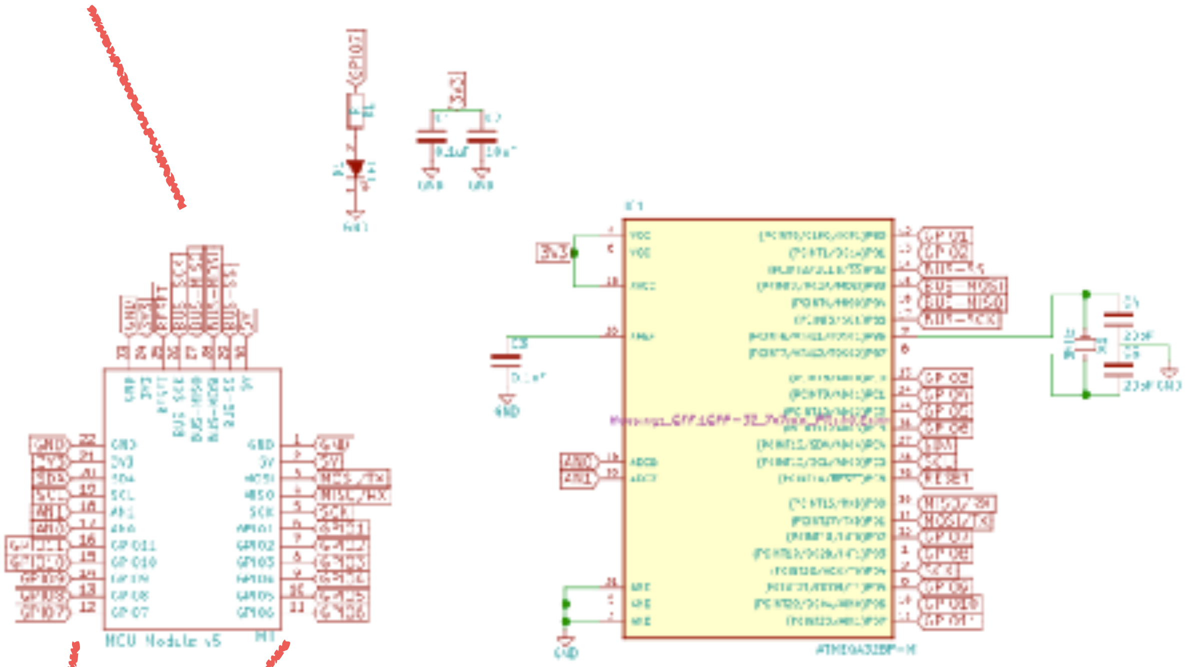
Abstract system block diagram:

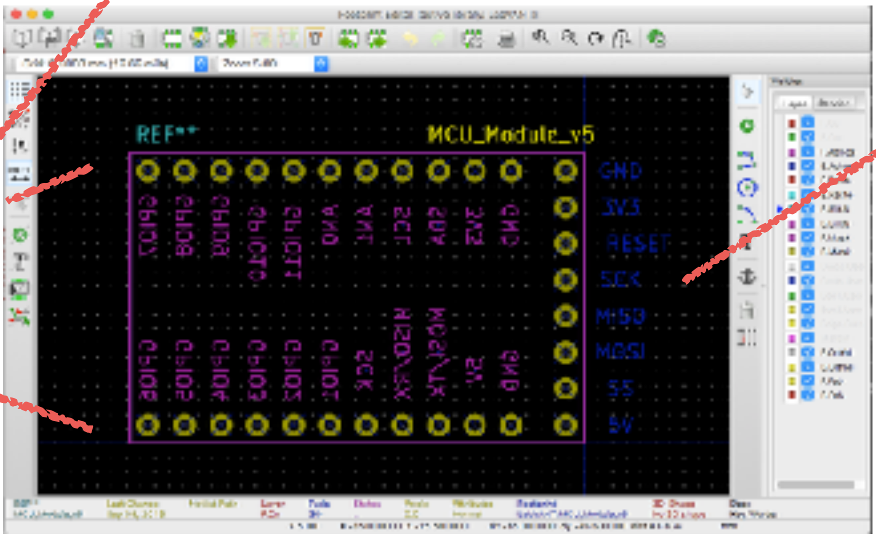
1. The system consists of a main bus that connects generic micro controller modules (MCU).
2. The bus carries SPI signals, 3.3V and 5V power lines and reset connections to each module.
3. Each MCU module has a main bus connection and side connections to connect to submodules.
4. Submodules have specific hardware installed.
5. One module connected to the main bus is a SPI master and is responsible for handling communications between all the other modules. This module is called the COM module.
6. The COM module's GPIO lines are used to connect to the other modules SS pins.



MCU module: an ATMega328P in a little PCB, connected to the other modules by a main SPI bus. This module is surrounded by pads much like an Arduino stamp. These pads form a standard connection to a smaller module which I call a submodule. They deliver a second SPI, an I2C, a serial connection, two analog inputs and some GPIOs to the submodule. To enable the second SPI I had to remove the boot loader from the ATMega328P and strip all Arduino and wiring traces, rewriting what I needed from the beginning and redefining many things.

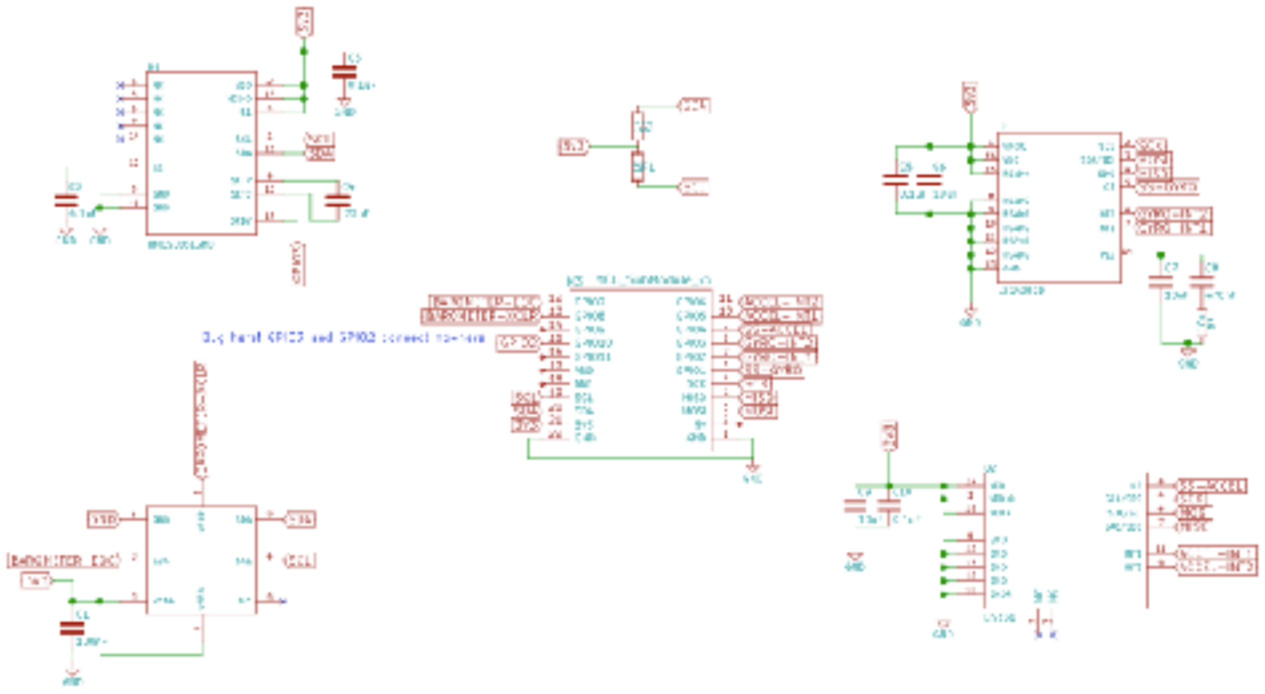






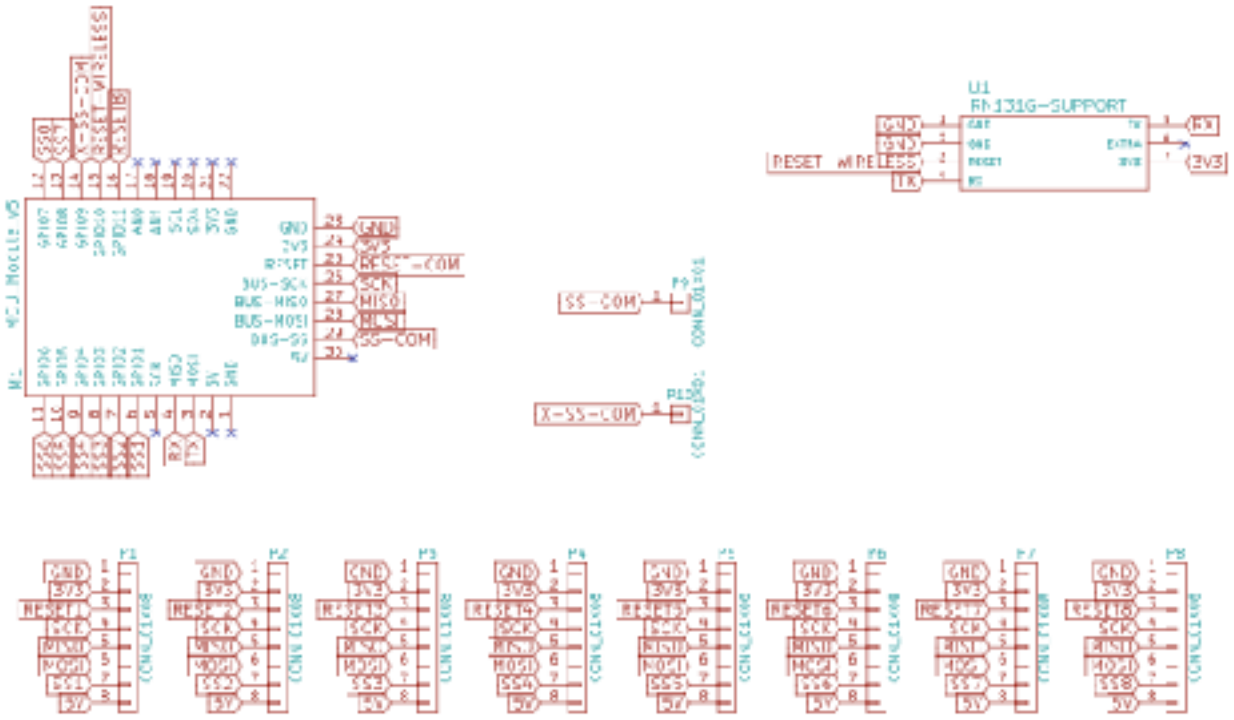
AMGP sensor submodule: a small board that connects to the MCU module. This submodule has an accelerometer, a gyro, a magnetometer and a barometer. They connect to the MCU module using I2C and SPI buses, both available on the lateral connections. Data is read by the MCU module, processed and made available for other modules when they need.

In this picture the accelerometer submodule is already connected to a MCU module.

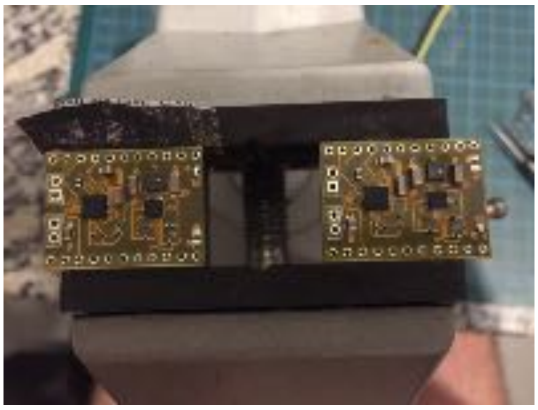


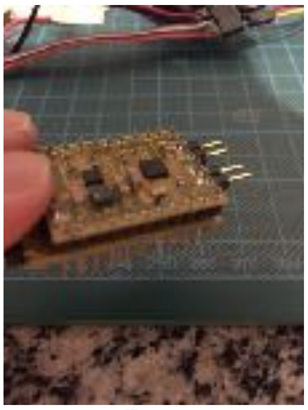
COM module: This is a standard MCU module also connected to the main SPI bus, but with connections to the slave select pins of the other modules. It is also connected via UART to a WIFI module that delivers the required information wirelessly to my computer. This module is the messenger. It calls other modules, identifies them and pulls all the available information, delivering everything later to the flight control module.

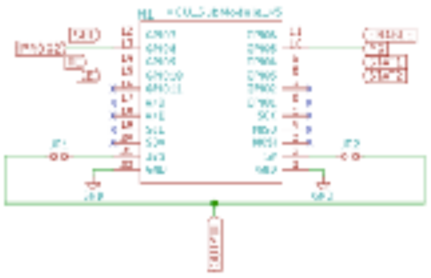
The Backbone: A PCB with an optimal form factor to plug the modules and host a RN131G WiFi and a COM module.

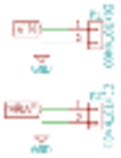


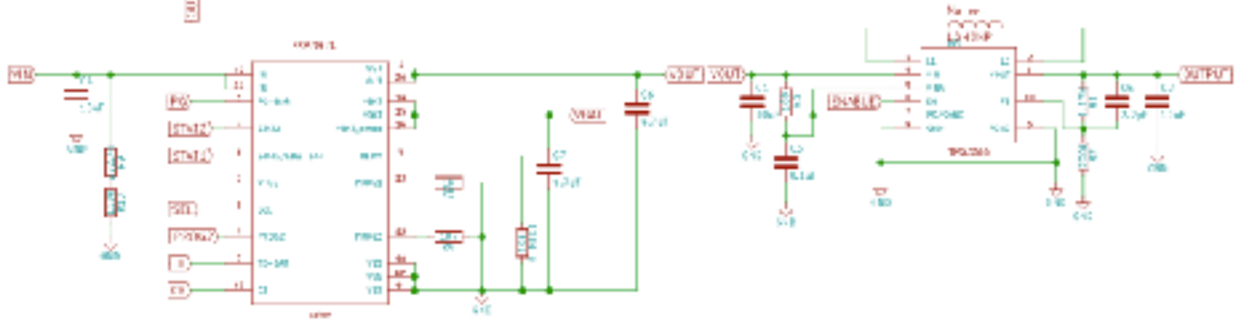
PS Modules: Two power supplies. Each one fit in a submodule and is able to charge a LiPo cell and power up the system with up to 0.8A at 5V and 1.4A at 3.3V. The PS submodule also connects to a MCU Module, which connects to the main bus at the Backbone PCB.



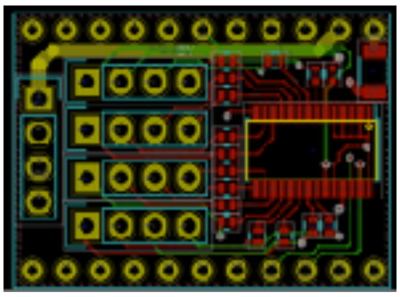


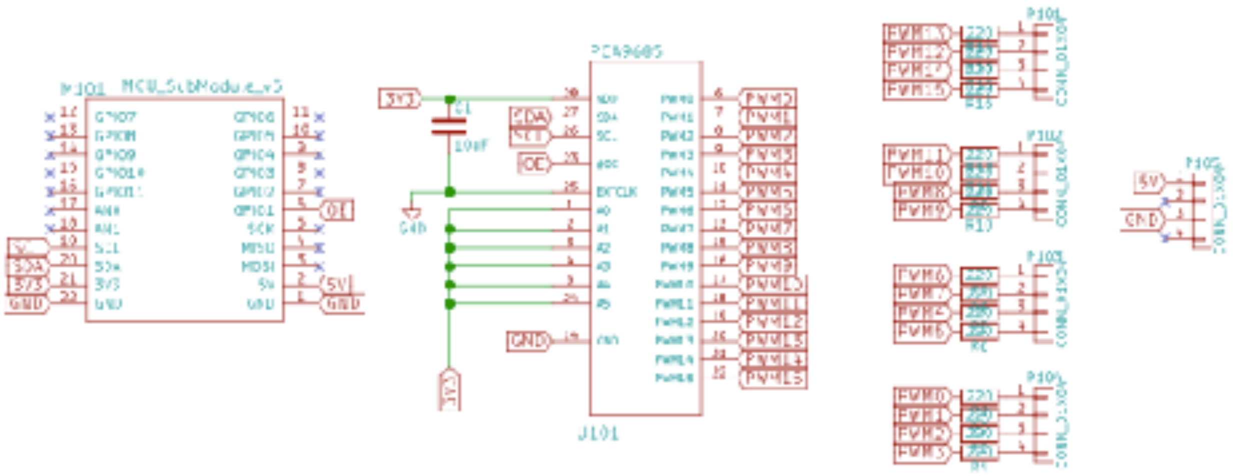






Servo control submodule: A submodule sized PCB which connects to a MCU module and can control up to 16 servos.




P101 MCU_SubModule_v3
PCA9685
J101
C1
10uF
3V3
SDA
SCL
OE
GND
5V
P101
P102
P103
P104
P105