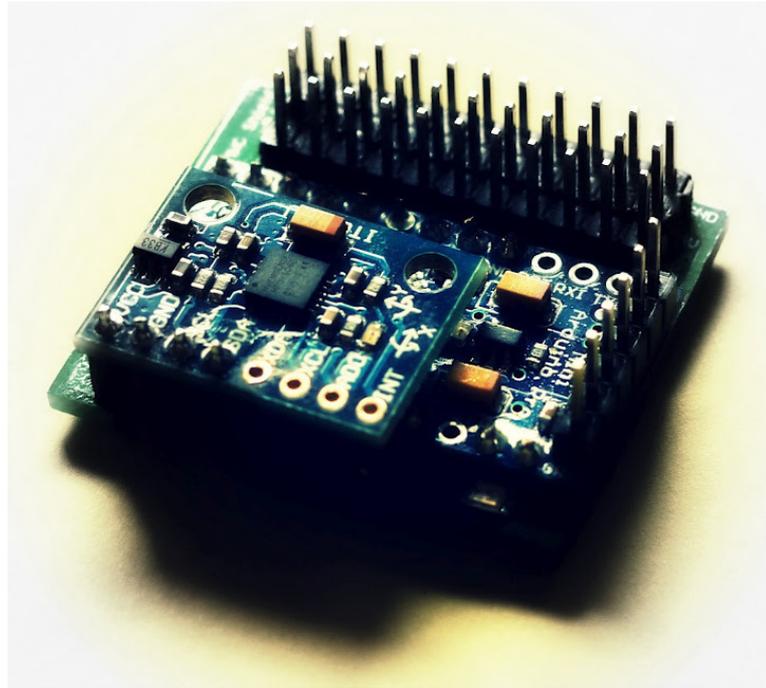


mMWC shield

mini MultiWiiController shield

optimized for airplane stabilisation



HowTo GUIDE

Introduction:

multiWii is open source software, mainly intended for multicopter use. Recently this great software supports airplanes as well. While the actual controlling part is arduino based the software supports a wide range of sensors. The principle of multiWii is to put a "flight controller" between the receiver and the motor/servos. This means, the pilot is not directly controlling the aircraft with the receiver impulses created by moving the transmitter sticks. Actually those received impulses are read by the multiwii controller and - depending on certain preconfigured modes & fine tuned control loops, movement/level of the aircraft, etc. - the flight controller will actually send controlling impulses to servos (and motor).

mMWC is a carrier board (shield) that provides a small footprint in order to fit into most even more narrow plane fuselages. Carrier board means, the board itself actually carries the controller and the sensors, while providing connection pins for receiver and servos.

mMWC has been reduced to the minimum required for airplane stabilisation with multiWii resulting in a light weight (8g) small (30x35mm) stabilisation system that allows similar like expensive commercial products different modes (no stabilisation, acro-mode, FPV-level-mode) to be selected with a switch on the transmitter.

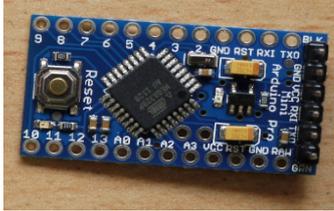
compared to much more expensive commercial solutions mMWC has the big advantage of running open source software that is continuously getting new features. e.g. recently for copters multiWii has support of GPS position hold and return-to-home. mMWC uses I2C-interface for the sensors and could also take advantage of I2C-GPS in future when multiwii supports that for planes as well...

What you need to stabilize your plane with mMWC:

- **Arduino Pro Mini 5V**

The actual “brain” of the flight controller.

clones with an Atmel 328P are quite cheap on ebay, approx. 10 USD



- **MPU 6050 IMU**

motion processing unit, a 6DOF IMU that combines Gyro and Accelerometers, providing I2C bus interface towards arduino.

Ensure to get a version with the following layout in order to be able to sandwich the sensor IMU with mMWC:

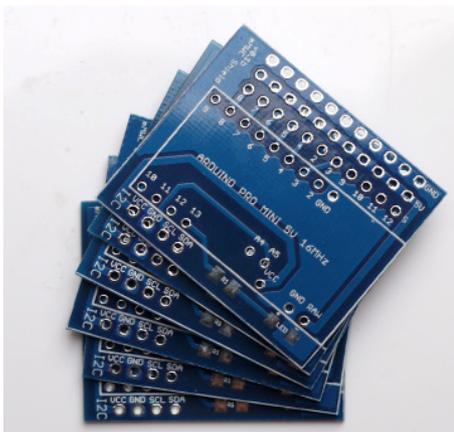


around 10 USD on ebay.

otherwise one can still wire the IMU with the mMWC and place it then somewhere in the plane

- **mMWC shield set**

comes as plain PCB, in nice cool dark blue 1mm thin material



also delivered with mMWC a bright blue SMD-LED (and suitable resistor) which can even be seen on sunny daylight. Note, that the LED/resistor can be used optional. mMWC will work without that as well. Coming with mMWC is a tiny bluetooth-to-flight controller adapter board. this can optionally be used to extend your mMWC sandwich by bluetooth (see next bullet).

can be ordered online see <http://www.rcgroups.com/forums/showthread.php?t=1684727> or <http://www.rcgroups.com/forums/showthread.php?t=1684736>

9.5 USD for one piece, 15 USD for 2 pieces incl. worldwide shipping

- **FTDI connector**

The arduino pro mini provides one serial interface. In order to configure/update the software, etc. a so-called FTDI-connector is used. If you are keen with arduino already, you most probably already own one FTDI-connector, if not, get one for cheap (either together with arduino pro mini, or separate) on ebay (around 7 USD)



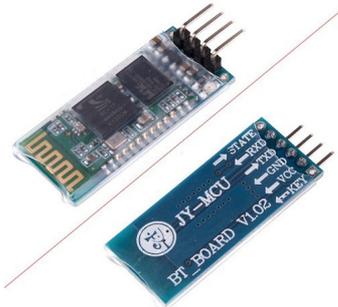
- **solder pin headers**

can be sourced locally quite easy



- **bluetooth dongle (optional)**

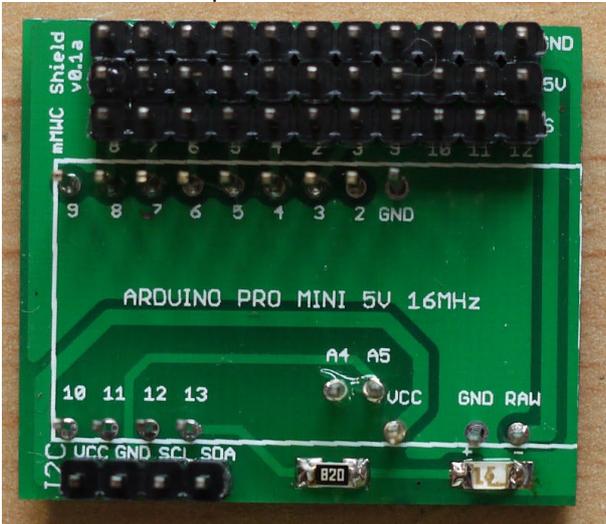
The arduino pro mini provides one serial interface. Adding bluetooth to multiwii flight controller allows you to use your computer or smartphone to connect wireless to your flight controller and check status/values of sensors, fine tune the PID loop, configure AUX switch functionality, etc...



get one on ebay for less than 9 USD.

How to build the mMWC:

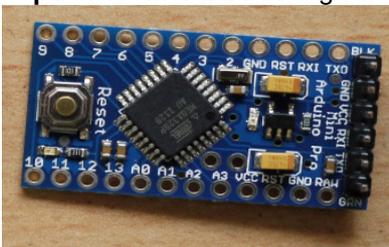
If you decide to use them, start with the SMD-LED and resistor, as in the beginning you have better space so that it does not get too tricky. For the resistor, there is no polarity to be concerned, but for the LED please use it exactly as shown on that picture:



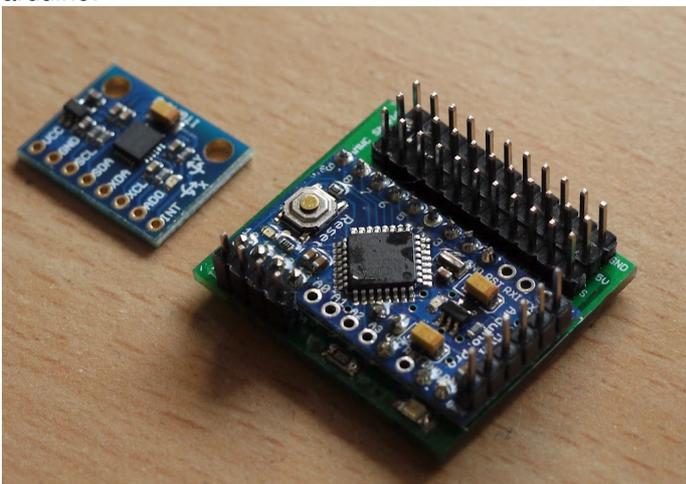
Next solder all the pins required.

Actually on the pin rows for receiver connection, GND and +5V is required only once, you could spare some weight if you omit GND and +5V on the remaining receiver channel inputs on the mMWC.

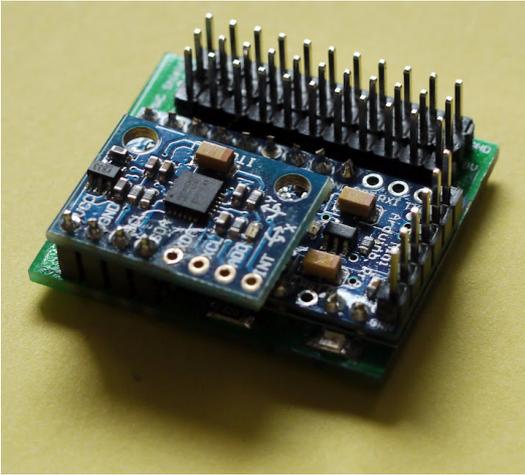
Important: before soldering the arduino pro mini on to the mMWC, ensure to have the serial port pins equipped:



Now solder the arduino pro mini on to the mMWC and cut the remaining piece of the pins that stay out on top of the arduino:



Last but not least, add the sensor IMU:



you should be done with all of the above in a couple of minutes and you will end up with a nice looking PCB sandwich like this:

Get the software:

- download and install Arduino 1.0.1 from <http://arduino.cc/> for your operating system. This is the SDK you will need for editing and compiling the multiwii software as well as for downloading the binary to the Arduino with the FTDI adapter.
- Download and extract MultiWii 2.1 from <http://code.google.com/p/multiwii/downloads/list> beside the release note you will find 2 directories:
 - MultiWii_2_1 contains the source files
 - MultiWiiConf_2_1 has the configuration application

Adjust MultiWii for mMWC:

MultiWii 2.1 supports many different setups in terms of microcontrollers, sensors, aircrafts. First thing to do is always to adjust and compile it for your specific configuration:

- start Arduino 1.0.1
- open MultiWii_2_1.ino from the extracted MultiWii archive
- The editor will open several files now, you should head to the config.h
This file contains all configuration parameters you need to adjust multiwii for mMWC.

configuration in that context here means in most cases to “enable” certain parameter values by uncommenting them.

1.) select airplane mode:

the very first section called “type of multicopter” contains a lot of different aircraft types.
find the line:

```
//#define AIRPLANE
```

and change it to:

```
#define AIRPLANE
```

2.) enable and configure MPU6050 orientation:

the section “independent sensors” lists lot of different sensors. Enable the MPU6050:

```
#define MPU6050 //combo + ACC
```

and configure it's orientation.

```
/* individual sensor orientation */  
#define ACC_ORIENTATION(X, Y, Z) {accADC[ROLL] = -X; accADC[PITCH] = -Y; accADC[YAW] = Z;}  
#define GYRO_ORIENTATION(X, Y, Z) {gyroADC[ROLL] = Y; gyroADC[PITCH] = -X; gyroADC[YAW] = -Z;}
```

3.) define AUX2 pin:

section 4 - alternate CPU & BOARDS has at the beginning of the section a line where you can enable the AUX2 port:
on the mMWC this is pin8:

```
#define RCAUXPIN8
```

4.) disable ARming via Yaw/Roll:

we do not want to enable/disable motor arming with sticks. instead we will use an aux-switch, therefore comment out those lines:

```
// #define ALLOW_ARM_DISARM_VIA_TX_YAW  
// #define ALLOW_ARM_DISARM_VIA_TX_ROLL
```

compile the code and upload the binary

the common basic settings for mMWC for airplane mode are done now.
To upload the software to the board follow these steps:

- select "Arduino Pro Mini 5V Atmega 328" in Arduino 1.0.1 under menu Tools/Board.
- Next plugin your USB-cable and connect it to the FTDI adapter.
- select the adapter under Tools/Serial Port
- connect the FTDI to your arduino. Note, there are 2 ways to connect. orientate on the labels printed on the arduino and on the FTDI. actually connecting it the wrong way did never fry any of my boards so far...
- you should see if the FTDI connection is fine, when any LED on the arduino goes active
- click the "upload" button (the one with the arrow pointing to the right) in the editor window. If everything runs well, you should get some message like "Upload done." or similar. If not, check FTDI connection and selected board/serial port again.

bench testing mMWC:

- start up multiwii config tool (java)
- select the COM-port for your FTDI
- click the start button

now verify the following: if you move/tilt the board into different directions that the roll/pitch arrows change their direction accordingly.

on the left bottom you can enable/disable each sensor axis extra.

put the mMWC on a level surface (desk) in front of you.

the blue LED should be in the left lower corner

start with acc roll and gyro roll:

tilt the board to the left and check the curves: both should decrease their value. well, actually after the movement is over and you keep the board in the left position, only the ACC value will remain at its lower value, while the gyro is back to neutral again.

When tilting to the right, things are the other way around: the curves will increase their value here.

Note: roll will control the ailerons lateron on our airplane.

pitch: tilt the board back to you and the values decrease, forward will increase.

Note: this will adjust our elevator on the airplane.

yaw: is measured on the MPU6050 by the gyro only. some might use magnetic compass sensor in addition to that as well, but the MPU6050 alone uses just the gyro.

keep the mMWC on a level surface and turn it clockwise, the curve should increase. counter clockwise turn will decrease the value.

Note: this will compensate the rudder lateron on our aircraft.

Ensure also, MultiWii Config GUI does not show any I2C errors.

Configure flying modes and ARming:

you should use at least AUX1 but better AUX2 also.

Configure AUX1 as the mode-switch:

MIN: passthrough (means, stabilisation turned off)
MED: - (means acro-mode, i.e. gyro only)
MAX: ACC (means level-mode, for FPV)

AUX2 can be used as an "ARM" switch.

When disarmed the plane (for example on the bench) will never run the motor.

Well, I put this on a safety switch on my Tx, so that I will not - by accident - disarm the motor during flight.

Putting all the stuff together...

1.) mount mMWC into your airplane

When mounting mMWC in your airplane ensure to reduce vibrations as much as possible.

Most important here is to have good balanced motors and props.

Mounting the mMWC with the help of some soft foamed rubber also will reduce the vibrations.

The place of the mMWC in your plane should be as close as possible to the CG (center of gravity) of your airplane.

Ensure that the orientation is correct (like before in the bench testing).

Also try to mount it as much level as possible, although actually we will calibrate the sensors later on anyway...

remove the prop from the motor for all the steps below.

2.) connect the servos and the ESC as described on the back of the mMWC:

Servo/ESC	mMWC pin
ESC/Motor	9
Rudder-Servo	3
Elevator Servo	10
Aileron Servo1	11
Aileron Servo2	12

3.) connect the receiver to the mMWC as described on the back of the mMWC:

Receiver channel	mMWC pin
Throttle	2
Aileron	4
Elevator	5
Rudder	6
Aux1	7
Aux2	8

4.) calibrate your transmitter:

the very first step with multiWii is always to calibrate the transmitter;

- turn on your transmitter
- connect battery to ESC
- connect FTDI with computer and Arduino
- start multiWii config GUI

The GUI shows for each receiver channels the current values. use your transmitter's calibration menu to end up with

1095 for MIN-values

1500 for middle values and

1905 for MAX-values

on all channels. this ensures that the multiWii-software lateron is able to detect certain thresholds.

For example, you will not be able to ARM your motor (using an AUX-switch) when the throttle value is above 1100.

IMPORTANT: After having the switches configured check in multiWii config GUI they act as expected.

Check, if as long as multiwii is disarmed the throttle stick will not cause the motor running and if you can proper arm the motor.

Check further if the different modes can be chosen. especially in pass-through mode any movement of the board should not cause any movement of any servo.

configure servo direction:

As each servo directions differ on airplane models, you can here tell Multwii the actual direction it should move a certain servo.

I never found so far a description about those fields, but the digging into the code, i found that (starting with 0) the positions 3 and 4 are meant to describe the aileron servos, 5 is the rudder and 6 the elevator.

in other words, ignore the first 3 values and the last one.

```
//  
#define SERVO_DIRECTION { -1, 1, 1, AIL1 AIL2 RUDD ELEV  
-1, 1, 1, 1, 1 }
```

While still having battery on the ESC, ensure to have level-FPV mode activated (in that case it is easier to see the direction the servos are moving)

move/tilt the plane into different directions that should be stabilized and check, if the servos move to the expected direction.

For any servo that moves the wrong way, invert the appropriate value (-1 to 1 or vice versa) in the SERVO_DIRECTION

array, compile, download and test again.

when you are done with all servos try if moving the sticks on your transmitter cause the servos to move to the right direction. if something moves wrong, correct the direction on the transmitter now.

calibrate the sensors

- **ACC calibration:** this needs to be done usually only once. ensure your plane is in perfect level, use MultiWii Config GUI to connect to the controller and click the calibrate ACC button. Done!
- **Gyro calibration:** this is done automatically every time you power up your flight controller. important for gyro calibration is that your aircraft does not move at that moment. it does not matter, if you have it laying upside down in the grass or something. Just connect the battery and wait a few seconds.

first flight with mMWC

- switch to acro-mode and check if the servos compensate correctly
- switch to FPV level mode and check if the servos compensate correctly
- Enable pass through mode and check your plane as you would do with any other plane before it's first flight.
- ensure none of the movements will change any of the servo movements
- Arm the motor and fly your plane to safe height.
- enable either acro-mode or FPV-mode and check it's behavior.
- If something acts strange immediatly switch back to pass-through mode and land your plane

fine tuning PID values

... to be continued ...

setting up Bluetooth to work with mMWC

... to be continued ...