

Đồ án 2: Báo cáo mô hình Quadcopter sử dụng Arduino R3 và module gyro MPU6050

I. Giới thiệu chung

1. Lịch sử phát triển



- Máy bay không người lái (viết tắt tiếng Anh: UAV - Unmanned aerial vehicle) là tên gọi chỉ chung cho các loại máy bay mà không có phi công ở buồng lái và được điều khiển từ xa từ trung tâm. Theo sự phát triển công nghệ hiện có các dạng UAV:

Máy bay theo nghĩa truyền thống được trang bị hệ thống điều khiển và lái tự động, được gọi là UAS (unmanned aircraft system), xuất hiện từ những năm 1950 và đã từng phục vụ việc do thám và trinh sát chiến trường. Loại tổ hợp máy bay này có khả năng tự động hóa các hoạt động của máy bay cao, không đòi hỏi những trang thiết bị hàng không đặc chủng, giá thành khai thác sử dụng và bảo trì hệ thống để phục vụ lâu dài rẻ, trong quân sự loại máy bay này có đặc tính tấn công chớp nhoáng.



Thiết bị bay kiểu mới, được chế tạo rất đa dạng, có kích thước và công suất động cơ nhỏ đến trung bình, được gọi là drone. Ứng dụng của UAV drone đang tăng lên mạnh mẽ, từ các mục đích quân sự cho đến nghiên cứu khoa học, điện ảnh - truyền hình, nông nghiệp, thương mại, giải trí.

Các drone có lắp camera để quan sát, và thường được gọi là flycam. Để thuận tiện điều khiển thao tác thì drone có nhiều cánh quạt, thường là 4.



- Thiết bị dùng cho quay phim chụp ảnh trên không (drone with camera) hay flycam (flying camera), là một loại thiết bị bay không người lái có lắp camera hay máy ảnh để quay phim hoặc chụp ảnh từ trên cao. Nó được điều khiển từ xa bằng trình điều khiển riêng biệt hoặc có thể kết nối vào điện thoại, máy tính bằng để điều khiển qua sóng wifi. Ngày nay các thiết bị này được sử dụng rộng rãi trong lĩnh vực quay phim, chụp ảnh từ trên cao .





- Tuy nhiên khi sử dụng các thiết bị này, chủ máy cần phải được cơ quan có thẩm quyền cấp phép sử dụng, đặc biệt là tại các địa điểm quan trọng về quân sự, chính trị, hoặc trên vùng đất công cộng... cũng như quan hệ với quyền riêng tư của cá nhân hay cộng đồng khác. Tại các nước phát triển thì có Quy định về sử dụng thiết bị bay không người lái (Regulation of unmanned aerial vehicles) rõ ràng.

Drone nông nghiệp là loại UAV có cả chức năng *flycam* để chụp ảnh và các cảm biến khác để quan sát môi trường, là ứng dụng UAV vào nông nghiệp để giám sát các trang trại rộng lớn.

2. Tiềm năng

- Một vài ứng dụng tiềm năng của máy bay không người lái trong cuộc sống hàng ngày đã được ông Quốc chia sẻ với người viết bài.

- Xin bắt đầu bằng nông nghiệp, một lĩnh vực quen thuộc. Trong trường hợp canh tác trên diện tích lớn, vài trăm héc ta trở lên chẳng hạn, việc theo dõi, kiểm tra sự phát triển của cây trồng không phải lúc nào cũng dễ dàng. Đặc biệt với những cây trồng có chiều cao ngang hoặc hơn đầu người, công việc còn phức tạp hơn bởi lẽ người nông dân đi vòng ngoài khó quan sát hết được tình hình phát triển thực tế của cây trồng bên trong.

Lúc này, bằng cách sử dụng drone gắn thêm camera quan sát, cho bay một vòng chụp hình ảnh của cánh đồng, ta sẽ có bức tranh tổng thể về tình hình phát triển của cây trồng. Từ đó, người quản lý có thể xác định rõ ràng vùng nào cây phát triển tốt, vùng nào phát triển kém để đưa ra giải pháp can thiệp kịp thời. Trong giai đoạn này, drone chỉ mới xác định được vùng cây trồng nào có dấu hiệu bất thường. Còn bất thường như thế nào, bị bệnh gì, vẫn cần cử nhân viên trực tiếp đến hiện trường để kiểm tra.

Theo thời gian, khi hệ thống quản lý trang trại có thêm nhiều dữ liệu, việc tích hợp thêm trí tuệ nhân tạo để tăng tính tự động hóa trong việc chăm sóc cây trồng là điều khả thi. Ví dụ, sau khi camera từ drone gửi ảnh thực tế nông trường về hệ thống, dựa trên nền tảng trí tuệ nhân tạo, hệ thống có thể tự động xử lý, chẩn đoán tình trạng bệnh mà cây trồng đang gặp và đề ra phương án điều trị. Từ phương án này, một drone khác có thể được điều khiển đến để phun thuốc bảo vệ thực vật hoặc rải thêm phân bón.

- Việc ứng dụng drone cũng cho thấy nhiều tiềm năng trong lĩnh vực phòng cháy chữa cháy (PCCC). Thực tế cuộc sống chỉ ra rằng có không ít đám cháy, tưởng đã dập tắt, nhưng sau đó lại bùng lên, thậm chí cháy mạnh hơn. Lý do là sau lần chữa cháy đầu tiên, hiện trường vẫn còn những ổ nhiệt, tuy không phát lửa nhưng sẵn sàng cháy khi gặp điều kiện thích hợp. Để đề phòng tái cháy, việc sử dụng drone gắn camera nhiệt có thể giúp phát hiện ra những ổ nhiệt để xử lý tận gốc.

Việc sử dụng drone kết hợp camera quan sát cũng đem lại nhiều thông tin hữu ích trong và sau quá trình xử lý đám cháy. Trong quá trình chữa cháy, đặc biệt tại các công trình cao tầng, nơi con người khó tiếp cận nguồn cháy, thông tin từ camera đưa về có thể giúp đơn vị PCCC xác định rõ đâu là điểm trọng tâm cần xử lý cũng như quan sát được bức tranh toàn cảnh. Nhờ vậy, việc chữa cháy hiệu quả hơn. Sau cùng, những hình ảnh ghi lại từ camera trong quá trình chữa cháy có thể giúp đơn vị liên quan xem lại và rút ra những bài học thiết thực.

Bên cạnh PCCC, drone còn hữu ích thiết thực trong việc cứu hộ cứu nạn. Xin thử hình dung, vì một sự cố bất ngờ chẳng hạn, một người sắp chết đuối giữa biển. Trong tình huống này, thay vì chờ tàu cứu nạn, một chiếc drone được điều khiển mang theo phao cứu sinh sẽ là giải pháp tiết kiệm thời gian hơn rất nhiều. Chi tiết này không đến từ trí tưởng tượng mà thực tế đã xảy ra vào cuối tháng 1 vừa qua tại Úc.

- Ngày nay chúng ta nghe nhiều đến năng lượng xanh. Nổi bật trong số đó là năng lượng gió và mặt trời. Với năng lượng gió, mỗi turbine gió cao đến vài chục mét; ví dụ như turbine tại nhà máy điện gió Bạc Liêu cao đến 80 mét. Việc kiểm tra cánh quạt hay các chi tiết trên cao, trong một số trường hợp, cần phải cho turbine ngưng hoạt động và đưa nhân viên lên. Cách làm này có nhược điểm nhất định.

Giờ đây, nếu dùng drone gắn camera quan sát tiếp cận điểm cần kiểm tra thì các nhược điểm trên có thể khắc phục. Hệ thống không phải ngưng hoạt động và cũng không cần thiết phải đưa người lên cao.

Câu chuyện tương tự cũng diễn ra với việc kiểm tra, kiểm định tầng mái các công trình xây dựng trên cao. Ví dụ bạn lắp hệ thống các tấm pin mặt trời trên nóc nhà chung cư. Vậy làm sao biết các tấm pin mặt trời có được lắp ngay hàng thẳng lối? Liệu quá trình lắp đặt có làm hỏng, vỡ, trầy xước các tấm pin? Cách làm truyền thống là đưa nhân viên lên, di chuyển một vòng, chụp hình rồi báo cáo.

- Nếu đưa drone vào sử dụng, thiết bị này có thể gánh bớt một phần việc cho con người. Điều này không chỉ giúp đẩy nhanh tốc độ kiểm tra mà còn giảm rủi ro cho nhân viên. Khi những ứng dụng này trở nên phổ biến, thị trường lao động sẽ xuất hiện một nghề mới - nghề điều khiển máy bay không người lái trong lĩnh vực dân dụng.

3. Ứng dụng thực tế

a. Quân sự

- Máy bay không người lái ngày càng được ứng dụng rộng rãi trong quân sự như vận chuyển đạn dược, thuốc men đến chiến tuyến nhanh chóng với địa hình khó khăn hiểm trở

- Biên giới các nước luôn được đặt ở mức bảo vệ nghiêm ngặt, tuy nhiên việc nhập cư và buôn lậu trái phép đang là vấn nạn của rất nhiều quốc gia trên thế giới, chính vì thế có thể kiểm soát được dọc biên giới là điều rất quan trọng. Và cũng như những đồng chí cảnh sát, các cơ quan biên phòng cũng đã tận dụng những **công dụng của máy bay điều khiển Drone** để tìm kiếm những mối nguy hiểm hoặc phát hiện những ai đang có ý định xâm phạm lãnh thổ của quốc gia.

Thay vì phải đi có người đi tuần tra và canh gác thì việc canh gác sẽ diễn ra hiệu quả hơn nhờ vào việc sử dụng thiết bị drone với góc nhìn rộng từ trên cao và khả năng di chuyển nhanh. Chính phủ Úc gần đây cũng đã mua rất nhiều drone để phục vụ cho mục đích này.



b. Kinh tế

- **Khảo sát công trình xây dựng, dàn khoan dầu khí**

Các nhà thầu, chủ đầu tư của những công trình xây dựng hay dàn khoan dầu khí thường luôn muốn có cái nhìn toàn cảnh và rõ nét những quy trình xây dựng hoàn thiện của mình, vậy nên thiết bị Drone là lựa chọn hợp lý nhất dành cho họ. Đặc biệt là những dàn khoan dầu khí ở ngoài biển, sẽ rất khó khăn cho những nhà nghiên cứu khảo sát và xem xét việc hoạt động của dàn khoan.



c. Dịch vụ

- **Chụp ảnh, quay phim giải trí từ trên cao**

Ứng dụng nổi bật nhất mà khi nhắc đến máy bay điều khiển Drone hay Flycam bất kì ai cũng có thể nhận biết, đó chính là tính năng chụp ảnh, quay phim từ trên cao. Phá bỏ và vượt trội lên so với phương thức chụp ảnh truyền thống, chụp ảnh bằng flycam đem lại người chụp và người xem những góc chụp mới lạ và độc đáo. Đã có rất nhiều cá nhân và đơn vị bắt kịp xu hướng và những yêu cầu của khách hàng phát triển kinh doanh dịch vụ quay phim bằng flycam và dịch vụ chụp ảnh bằng flycam. Công dụng của máy bay điều khiển Drone đem lại cho những người có đam mê với công nghệ và nghệ thuật quay phim, chụp ảnh những shot hình hay thước phim vô cùng độc đáo mà không có bất kì thiết bị quay phim, chụp ảnh truyền thống nào có thể tạo được.



- **Shipper giao thức ăn, đồ tạp hoá từ trên cao**

Vào thời buổi công nghệ hiện đại hiện nay bạn chỉ cần ngồi nhà, lướt qua những trang web có đồ ăn hay đồ dùng bạn cần mua, nhắc điện thoại gọi và chỉ trong vài ngày hoặc thậm chí vài giờ sau bạn đã có trong tay đồ bạn cần thông qua cách vận chuyển nhanh mà giới trẻ thời nay thường hay gọi là shipper. Tuy nhiên, song song với sự phát triển hiện đại thì nhu cầu về chất lượng cũng như phương thức phục vụ của khách hàng cũng ngày càng tăng cao. Chính vì vậy, đã và đang có rất nhiều nhà kinh doanh nghiên cứu đến việc sử dụng thiết bị Drone làm công cụ vận chuyển hàng hoá. Công dụng của máy bay điều khiển Drone này hứa hẹn sẽ đem lại cho các nhà kinh doanh ngành bán lẻ rất nhiều lợi ích như tiết kiệm chi phí, công sức vận chuyển và bên cạnh đó cũng tiết kiệm được thời gian và tiền bạc cho người mua.

- Hỗ trợ cho ngành báo chí, truyền thông



Gần đây, có rất nhiều thông tin về những trường hợp phóng viên đang đi chiến trường hoặc khảo sát những địa hình hiểm trở gặp nạn trong lúc làm việc. Những thông tin đó gây hoang mang rất nhiều cho giới báo chí và truyền thông vì sự nguy hiểm mà công việc này đem đến. Ngoài những tai nạn nghề nghiệp nói trên, những phóng viên khi đang tác nghiệp cũng có thể sẽ bị bắt cóc, bị giết hoặc gặp nhiều rủi ro khác.

Tuy nhiên, trong thời buổi công nghệ ngày một tiên tiến và phát triển, thay vì phải tự mình thâm nhập vào những khu chiến sự, leo trèo trên những địa hình hiểm trở, thì những nhà báo, phóng viên trong tương lai có thể sử dụng thiết bị máy bay điều khiển Drone để ghi lại những hình ảnh, thước phim cần thiết mà không phải hi sinh và đối mặt với những rủi ro về mạng sống và thiết bị. Công dụng của máy bay điều khiển Drone này không chỉ dừng lại tại đó, những góc chụp, góc nhìn được thực hiện bằng thiết bị Drone có thể đem lại cho phóng viên, nhà báo cái nhìn toàn cảnh về vị trí nơi họ đang quan sát.

Đã có rất nhiều trường báo trí trên thế giới hiện nay đã bắt đầu giảng dạy cho sinh viên về việc sử dụng và những tiện ích của Drone để tác nghiệp.

II. CẤU TẠO VÀ NGUYÊN LÝ HOẠT ĐỘNG CỦA QUADCOPTER

1. Cấu tạo cơ bản



Về cơ bản, drone gồm các thành phần chính: vi mạch tích hợp bộ xử lý, động cơ, nguồn cấp năng lượng (pin), cánh quạt hoặc cánh bay. Điều khiển bay bằng bộ điều khiển từ xa hoặc/và lập trình sẵn theo lộ trình, tọa độ dựa trên GPS. Nhiều drone hiện nay đã được tích hợp GPS nên luôn định vị được đang bay ở đâu.

Bộ điều khiển drone thường sử dụng sóng radio tần số 2,4GHz, hình thức không khác mấy so với những bộ điều khiển từ xa của máy bay mô hình truyền thống, gồm hai nút bấm và ăng ten có thể gấp gọn. Một số bộ điều khiển có sự kết hợp cả tín hiệu 2,4GHz và Wi-Fi, trông giống tay cầm điều khiển máy chơi game hoặc chúng có thể dựa trên ứng dụng điều khiển chạy trên smartphone hay máy tính bảng.

Nhiều drone bay được là nhờ những cánh quạt quay, năng lượng do pin cung cấp. Tuy nhiên, những mẫu cao cấp đắt tiền có thể dùng động cơ phản lực, chúng có thể bay xa tới 800 km, và cao tới 15 km. Một số drone hoạt động nhờ được lập trình trước, số khác hoạt động dưới sự điều khiển từ xa của con người. Không như máy bay mô hình bay lượn cho vui, drone nói chung bay có đích đến, dù chúng được sử dụng cho mục đích quân sự hay dân dụng.

Những drone tiên tiến khi bay vượt tầm điều khiển, mất liên lạc, vẫn có thể bay theo đúng lộ trình lập sẵn, và tự quay trở về vùng có thể điều khiển.

Để bay lâu trên không, drone có thiết kế nhẹ. Tiện dụng, tính cơ động cao, hoạt động hiệu quả, thiết thực là những đặc tính nổi bật của drone. Những bộ vi xử lý ngày càng nhỏ và mạnh, phương thức liên lạc không dây được cải tiến không ngừng, cùng các loại cảm biến đa dạng đã mở ra cơ hội cho drone phát triển mạnh với nhiều ứng dụng hữu ích hơn.

Drone có thể phân thành hai loại chủ yếu, cánh cố định và cánh quạt. Mỗi loại đều có ưu nhược điểm riêng. Drone cánh cố định có thể cần đường băng để chạy lấy đà cất cánh, hoặc thậm chí phải dùng máy phóng. Ưu điểm của chúng là bay nhanh và lâu hơn loại cánh quạt. Trong khi đó, Drone cánh quạt phổ biến hơn do một phần dễ điều khiển, bay ổn định thích hợp cho nhiều hoạt động như chụp ảnh chẳng hạn. Đây là một nhu cầu rất lớn của người dùng khắp nơi chứ không riêng gì giới nhiếp ảnh gia chuyên nghiệp. Nếu bạn muốn tìm mua một drone cánh quạt, sẽ có rất nhiều lựa chọn trên thị trường dân dụng.

Cũng là thiết bị bay không người lái, vậy drone dân dụng khác gì máy bay mô hình truyền thống điều khiển từ xa qua sóng radio?

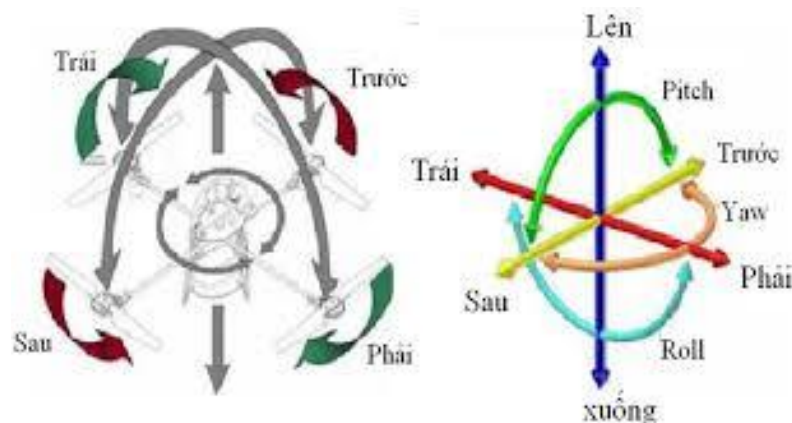
Về cơ bản, máy bay mô hình chủ yếu là để bay chơi, trong tầm quan sát của người điều khiển và đòi hỏi sự chú ý liên tục. Với drone, chẳng những điều đó không cần thiết mà nhờ khả năng tự động hóa cao, được trang bị những tính năng kỹ thuật số tiên tiến nên thực hiện được nhiều chức năng đáng kinh ngạc. Chẳng hạn lập trình định vị sẵn một vị trí GPS, drone có thể tự bay tới và "quần quanh" ở đó, tự bay trở về không cần điều khiển. Hoặc có thể thiết lập chế độ tự động bay theo để drone bám sát bạn từ trên cao nhờ tín hiệu dẫn đường phát đi từ smartphone của bạn mà nó thu nhận được.

Máy bay mô hình chỉ có một cánh quạt, nhưng drone dân dụng loại cánh quạt thường trang bị từ 3 – 8 cánh quạt. Drone cần nhiều cánh quạt để bay ổn định với sức nặng tổng cộng bao gồm cả những thứ mà nó phải mang theo để thực hiện nhiệm vụ được giao.

Các cánh quạt quay đẩy luồng khí xuống tạo ra phản lực từ bên dưới máy bay, vì thế thêm nhiều cánh quạt sẽ tăng sức nâng, thiết bị nhờ đó sẽ bay cao hơn, nhanh hơn và mang được trọng lượng nặng hơn – yếu tố quan trọng để drone mang thêm những thứ cần thiết như camera hay hàng hóa. Sức nâng yếu thì drone sẽ khó mà bay ổn định, thậm chí không thể bay nổi.

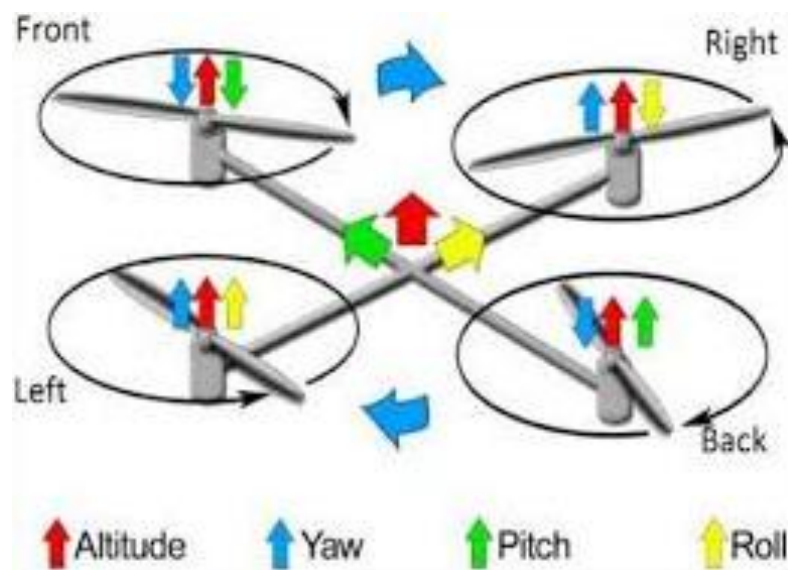
Một trở ngại với drone hiện nay là vấn đề năng lượng pin. Drone cần pin cấp nguồn cho động cơ. Cân bằng giữa công suất và trọng lượng của pin là một bài toán nan giải. Pin nhẹ thì thời lượng sử dụng ngắn mà tăng thời lượng pin sẽ tăng trọng lượng đồng nghĩa với tiêu tốn năng lượng khi bay.

2. Nguyên lí hoạt động



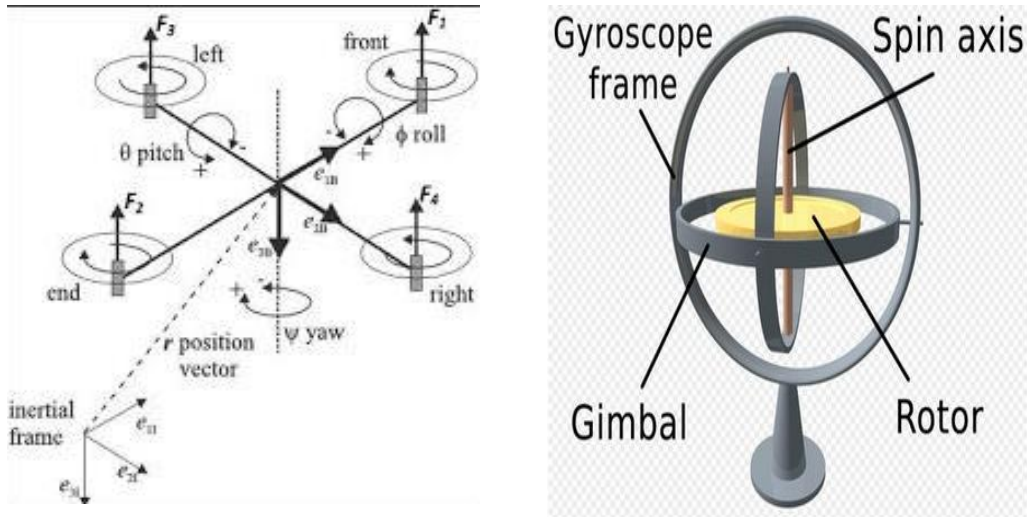


- Cặp cánh quạt phía trước (front) và phía sau (back) quay ngược chiều kim đồng hồ, trong khi đó cặp cánh bên phải (right) và bên trái (left) lại quay thuận chiều kim đồng hồ nhằm cân bằng moment xoắn được tạo ra bởi các cánh quạt trên khung. Cả 4 cánh phải sinh ra một lực đẩy bằng nhau khi Quadrocopter cất cánh và hạ cánh (throttle up/down). Góc xoay (roll) được điều khiển bằng cách thay đổi tốc độ giữa cánh bên phải và bên trái sao cho vẫn giữ nguyên tổng lực đẩy sinh ra bởi cặp cánh này. Tương tự như vậy, góc nghiêng (pitch) được điều khiển bằng thay đổi tốc độ của 2 cánh phía trước và phía sau mà vẫn giữ nguyên tổng lực đẩy. Trong khi đó, góc lệch (yaw) được điều khiển nhờ vào sự thay đổi tốc độ của cặp cánh phải – trái so với tốc độ của cặp cánh trước–sau mà tổng lực đẩy 4 cánh vẫn không đổi để Quadrocopter giữ được độ cao (Hình 1).



Hình 1. Chuyển động cơ bản của drone

Nguyên lý hoạt động chính của mô hình này hoạt động dựa trên sự chuyển động của các dòng khí do cánh máy bay tạo ra di chuyển xuống dưới làm vật bay lên trên và sự điều chỉnh vận tốc từng động cơ sẽ làm thay đổi hướng bay của Quadcopter. Để mô tả các chuyển động của một khung cứng 6 bậc tự do cần 2 hệ quy chiếu (Hình 2):



Hình 2. Hệ quy chiếu và gyro của MPU 6050


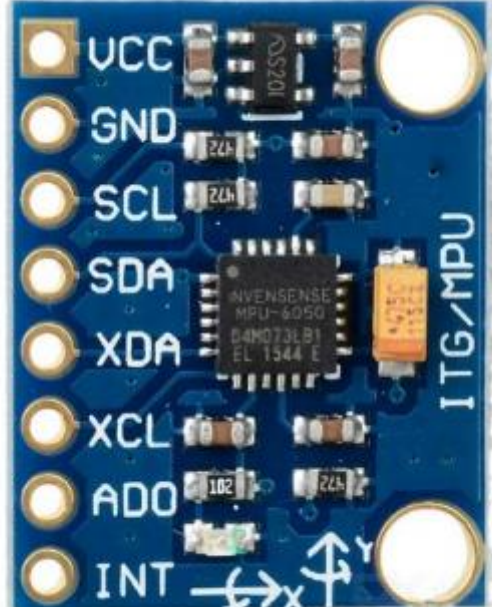
III. Thành phần và quy trình lắp ráp mô hình thực tế.

1. Thành phần cấu tạo



STT	Tên linh kiện	Hình ảnh thực tế
1	Khung thân quadcopter	
2	ESC Simonk 20 A	
3	Cánh quạt 10x 4,5R	
4	Motor 2212 1000kv	

5	Sạc 2- 3S	
6	Bộ thu phát RF	
7	Bộ đo dung lượng pin	
8	Battery (Pin)	

9	Arduino UNO	 The image shows an Arduino UNO R3 microcontroller board. It is a blue printed circuit board (PCB) with various electronic components. Key features include a USB Type-B port on the left, a DC power jack, a reset button, and a microcontroller chip (ATmega328P) in the center. The board has two rows of pin headers: a 26-pin Digital pin header at the top and a 6-pin Analog pin header at the bottom. Various components like resistors, capacitors, and a crystal oscillator are visible on the board.
10	Module MPU6050	 The image shows an MPU6050 module, which is a small blue PCB. It features a central black integrated circuit (IC) labeled 'MPU-6050'. The module has several pin headers on the left side, labeled UCC, GND, SCL, SDA, XDA, XCL, ADO, and INT. There are also two circular gold-colored pads on the right side, labeled 'ITG/MPU'. Various passive components like resistors and capacitors are soldered onto the board.

2. Chức năng từng thành phần

a. Khung thân quadcopter



- Sử dụng kit F450 tăng khả năng va đập và độ chắc chắn cho hệ thống.
- Được làm bằng nhựa siêu bền .

b. ESC



- **ESC:** là Electronic Speed Control: bộ điều tốc, như vậy ta cũng biết công dụng của nó rồi. Đây là thiết bị mạch điện giúp phân bố tăng hay giảm điện áp cho động cơ: điện tăng động cơ sẽ quay mạnh và ngược lại. Những loại mô hình thường dùng bộ điều tốc chính là những mô hình điện như: xe, máy bay, tàu,....

c. Cánh quạt



d. Motor



e. Sạc 3S



f. Bộ thu phát RF



Radio (Tx): thực chất ra nó còn có tên là **Transmitter** (cái này hông nhớ rõ là **Transmit** gì nữa, có gì anh em bỏ qua), bộ phát tín hiệu, người điều khiển sẽ cầm bộ điều khiển này để điều khiển mô hình thông qua sự nhận lệnh và "biên dịch" dòng lệnh của Rx. Bộ điều khiển càng nhiều "nút" (chức năng) và nhiều kênh thì càng mắc. Một vài bộ điều khiển cao cấp còn có cả màn hình tinh thể lỏng để xem tình trạng hoạt động của cả "đôi bên". Có hai loại dạng chính là dạng ngang và dạng súng. Dạng súng thì thường dùng cho tàu và xe, dạng ngang thì dùng cho máy bay.

Reciever (Rx): đây là một bộ phận thu sóng được đặt trong các loại mô hình dùng để nhận lệnh từ bộ điều khiển. Thiết bị này sau khi nhận lệnh, sẽ đưa ra một lệnh đến các thiết bị khác. Chẳng hạn như servo. Bộ thu nhận tín hiệu này có nhiều loại khác nhau dành cho từng mô hình khác nhau. Kênh ở đây là sự quyết định giá cả của bộ thu này. Càng nhiều kênh càng mắc, càng nhiều chức năng.

g. Battery



- Sử dụng pin lipo 1500mAh 3s 11,1V
- Hiệu suất dòng xả cao cung cấp cho hệ thống

h. Arduino UNO



- Sử dụng chip AVR ATmega328 của ATmel. Mạch arduino được lắp ráp từ các linh kiện dễ tìm và hướng đến đối tượng người dùng đa dạng. Đừng lo nếu bạn là dân nghiệp dư vì arduino có một hệ thống thư viện phong phú, cộng đồng người dùng arduino đông đảo sẵn sàng chia sẻ kiến thức và mã nguồn sẽ giúp bạn tạo nên những dự án thiết thực.

i. Module MPU 6050



➤ MPU-6050 tích hợp 6 trục cảm biến bao gồm:

- + con quay hồi chuyển 3 trục (3-axis MEMS gyroscope)
- + cảm biến gia tốc 3 chiều (3-axis MEMS accelerometer)

Ngoài ra, MPU-6050 còn có 1 đơn vị tăng tốc phần cứng chuyên xử lý tín hiệu (Digital Motion Processor - DSP) do cảm biến thu thập và thực hiện các tính toán cần thiết. Điều này giúp giảm bớt đáng kể phần xử lý tính toán của vi điều khiển, cải thiện tốc độ xử lý và cho ra phản hồi nhanh hơn. Đây chính là 1 điểm khác biệt đáng kể của MPU-6050 so với các cảm biến gia tốc và gyro khác.

MPU-6050 có thể kết hợp với cảm biến từ trường (bên ngoài) để tạo thành bộ cảm biến 9 góc đầy đủ thông qua giao tiếp I2C.

Các cảm biến bên trong MPU-6050 sử dụng bộ chuyển đổi tương tự - số (Analog to Digital Converter - ADC) 16-bit cho ra kết quả chi tiết về góc quay, tọa độ... Với 16-bit bạn sẽ có $2^{16} = 65536$ giá trị cho 1 cảm biến.

Tùy thuộc vào yêu cầu của bạn, cảm biến MPU-6050 có thể hoạt động ở chế độ tốc độ xử lý cao hoặc chế độ đo góc quay chính xác (chậm hơn). MPU-6050 có khả năng đo ở phạm vi:

+ con quay hồi chuyển: $\pm 250 \ 500 \ 1000 \ 2000$ dps

+ gia tốc: $\pm 2 \pm 4 \pm 8 \pm 16g$.

3. Quy trình lắp ráp.



- Lắp ráp mặt bottom cho khung cho mô hình



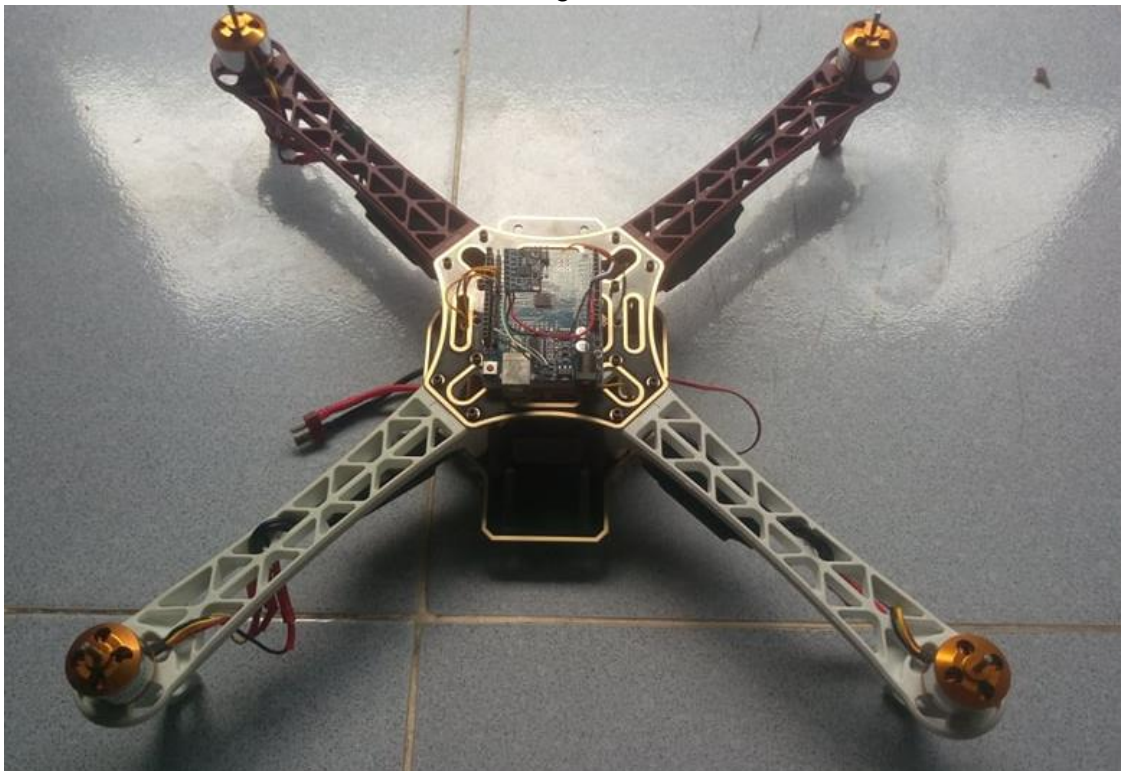
- Tiếp tục lắp phần top cho khung



- Lắp Motor và ESC vào khung quadcopter



- Kết nối ESC vào motor và PDB của khung



- Kết nối Arduino, receiver và MPU 6050 như sơ đồ bên dưới:

- **ESC Connections**

D3 << ESC 1 Signal Pin

D9 << ESC 3 Signal Pin

D10 << ESC 2 Signal Pin

D11 << ESC 4 Signal Pin

- **MPU-6050 Connections**

A4 << SDA

A5 << SCL

- **Receiver Connections**

D2 << Throttle

D4 << Elerons

D5 << Ailerons

D6 << Rudder

D7 << AUX 1



4. Code và phần mềm hỗ trợ

a. Code set up kênh và config của drone.

- `#ifndef CONFIG_H_`
- `#define CONFIG_H_`
-
- `/******`
`*****/`
- `/**** CONFIGURABLE PARAMETERS`
`*****/`
- `/******`
`*****/`
-
- `/* this file consists of several sections`

- * to create a working combination you must at least make your choices in section 1.
- * 1 - BASIC SETUP - you must select an option in every block.
- * this assumes you have 4 channels connected to your board with standard ESCs and servos.
- * 2 - COPTER TYPE SPECIFIC OPTIONS - you likely want to check for options for your copter type
- * 3 - RC SYSTEM SETUP
- * 4 - ALTERNATE CPUs & BOARDS - if you have
- * 5 - ALTERNATE SETUP - select alternate RX (SBUS, PPM, etc.), alternate ESC-range, etc. here
- * 6 - OPTIONAL FEATURES - enable nice to have features here (FlightModes, LCD, telemetry, battery monitor etc.)
- * 7 - TUNING & DEVELOPER - if you know what you are doing; you have been warned
- * - (ESCs calibration, Dynamic Motor/Prop Balancing, Diagnostics,Memory savings.....)
- * 8 - DEPRECATED - these features will be removed in some future release
- */
-
- /* Notes:
- * 1. parameters marked with (*) in the comment are stored in eeprom and can be changed via serial monitor or LCD.
- * 2. parameters marked with (**) in the comment are stored in eeprom and can be changed via the GUI
- */
-
-
-
- /*****
******/
- /*****
******/
- /***** SECTION 1 - BASIC SETUP
******/
- /*****
******/
- /*****
******/
- /*****
******/
-
- /***** The type of multicopter
******/
- //define GIMBAL

- `//#define BI`
- `//#define TRI`
- `//#define QUADP`
- `#define QUADX`
- `//#define Y4`
- `//#define Y6`
- `//#define HEX6`
- `//#define HEX6X`
- `//#define HEX6H // New Model`
- `//#define OCTOX8`
- `//#define OCTOFLATP`
- `//#define OCTOFLATX`
- `//#define FLYING_WING`
- `//#define VTAIL4`
- `//#define AIRPLANE`
- `//#define SINGLECOPTER`
- `//#define DUALCOPTER`
- `//#define HELI_120_CCPM`
- `//#define HELI_90_DEG`
-
- `/****** Motor minthrottle`
`******/`
- `/* Set the minimum throttle command sent to the ESC (Electronic Speed`
`Controller)`
- `This is the minimum value that allow motors to run at a idle speed */`
- `//#define MINTHROTTLE 1300 // for Turnigy Plush ESCs 10A`
- `//#define MINTHROTTLE 1120 // for Super Simple ESCs 10A`
- `//#define MINTHROTTLE 1064 // special ESC (simonk)`
- `//#define MINTHROTTLE 1050 // for brushed ESCs like ladybird`
- `#define MINTHROTTLE 1150 // (*) (**)`
-
- `/****** Motor maxthrottle`
`******/`
- `/* this is the maximum value for the ESCs at full power, this value can be`
`increased up to 2000 */`
- `#define MAXTHROTTLE 1850`
-
- `/****** Mincommand`
`******/`
- `/* this is the value for the ESCs when they are not armed`
- `in some cases, this value must be lowered down to 900 for some specific`
`ESCs, otherwise they failed to initiate */`

- `#define MINCOMMAND 1000`
-
- `/****** I2C speed for old WMP config
(useless config for other sensors) *****/`
- `#define I2C_SPEED 100000L //100kHz normal mode, this value must be
used for a genuine WMP`
- `//#define I2C_SPEED 400000L //400kHz fast mode, it works only with
some WMP clones`
-
- `/****** Internal i2c Pullups
*****/`
- `/* enable internal I2C pull ups (in most cases it is better to use external
pullups) */`
- `//#define INTERNAL_I2C_PULLUPS`
-
- `/****** constant loop time
*****/`
- `#define LOOP_TIME 2800`
-
-
- `/******
*****/`
- `/****** boards and sensor definitions
*****/`
-
- `/******
*****/`
-
- `/****** Combined IMU Boards
*****/`
- `/* if you use a specific sensor board:
please submit any correction to this list.`
- `Note from Alex: I only own some boards, for other boards, I'm not sure,
the info was gathered via rc forums, be cautious */`
- `//#define FFIMUv1 // first 9DOF+baro board from Jussi, with
HMC5843 <- confirmed by Alex`
- `//#define FFIMUv2 // second version of 9DOF+baro board from Jussi,
with HMC5883 <- confirmed by Alex`
- `//#define FREEIMUv1 // v0.1 & v0.2 & v0.3 version of 9DOF board
from Fabio`
- `//#define FREEIMUv03 // FreeIMU v0.3 and v0.3.1`
- `//#define FREEIMUv035 // FreeIMU v0.3.5 no baro`

- `//#define FREEIMUv035_MS // FreeIMU v0.3.5_MS`
<- confirmed by Alex
- `//#define FREEIMUv035_BMP // FreeIMU v0.3.5_BMP`
- `//#define FREEIMUv04 // FreeIMU v0.4 with MPU6050, HMC5883L, MS561101BA`
<- confirmed by Alex
- `//#define FREEIMUv043 // same as FREEIMUv04 with final MPU6050 (with the right ACC scale)`
- `//#define NANOWII // the smallest multiwii FC based on MPU6050 + pro micro based proc` <- confirmed by Alex
- `//#define PIPO // 9DOF board from erazz`
- `//#define QUADRINO // full FC board 9DOF+baro board from witespy with BMP085 baro` <- confirmed by Alex
- `//#define QUADRINO_ZOOM // full FC board 9DOF+baro board from witespy second edition`
- `//#define QUADRINO_ZOOM_MS// full FC board 9DOF+baro board from witespy second edition` <- confirmed by Alex
- `//#define ALLINONE // full FC board or standalone 9DOF+baro board from CSG_EU`
- `//#define AEROQUADSHIELDv2`
- `//#define ATAVRSBIN1 // Atmel 9DOF (Contribution by EOSBandi).`
requires 3.3V power.
- `//#define SIRIUS // Sirius Navigator IMU`
<- confirmed by Alex
- `//#define SIRIUSGPS // Sirius Navigator IMU using external MAG on GPS board` <- confirmed by Alex
- `//#define SIRIUS600 // Sirius Navigator IMU using the WMP for the gyro`
- `//#define SIRIUS_AIR // Sirius Navigator IMU 6050 32U4 from MultiWiiCopter.com` <- confirmed by Alex
- `//#define SIRIUS_AIR_GPS // Sirius Navigator IMU 6050 32U4 from MultiWiiCopter.com with GPS/MAG remote located`
- `//#define SIRIUS_MEGA v5_OSD // Paris_Sirius™ ITG3050,BMA280,MS5611,HMC5883,uBlox`
<http://www.Multiwiicopter.com> <- confirmed by Alex
- `//#define MINIWII // Jussi's MiniWii Flight Controller`
<- confirmed by Alex
- `//#define MICROWII // MicroWii 10DOF with ATmega32u4, MPU6050, HMC5883L, MS561101BA from http://flyduino.net/`
- `//#define CITRUSv2_1 // CITRUS from qcrc.ca`
- `//#define CHERRY6DOFv1_0`
- `//#define DROTEK_10DOF // Drotek 10DOF with ITG3200, BMA180, HMC5883, BMP085, w or w/o LLC`

- `//#define DROTEK_10DOF_MS // Drotek 10DOF with ITG3200, BMA180, HMC5883, MS5611, LLC`
- `//#define DROTEK_6DOFv2 // Drotek 6DOF v2`
- `//#define DROTEK_6DOF_MPU // Drotek 6DOF with MPU6050`
- `//#define DROTEK_10DOF_MPU//`
- `//#define MONGOOSE1_0 // mongoose 1.0`
`http://store.ckdevices.com/`
- `//#define CRIUS_LITE // Crius MultiWii Lite`
- `//#define CRIUS_SE // Crius MultiWii SE`
- `//#define CRIUS_SE_v2_0 // Crius MultiWii SE 2.0 with MPU6050, HMC5883 and BMP085`
- `//#define OPENLRSv2MULTI // OpenLRS v2 Multi Rc Receiver board including ITG3205 and ADXL345`
- `//#define BOARD_PROTO_1 // with MPU6050 + HMC5883L + MS baro`
- `//#define BOARD_PROTO_2 // with MPU6050 + slave MAG3110 + MS baro`
- `//#define GY_80 // Chinese 10 DOF with L3G4200D ADXL345 HMC5883L BMP085, LLC`
- `//#define GY_85 // Chinese 9 DOF with ITG3205 ADXL345 HMC5883L LLC`
- `//#define GY_86 // Chinese 10 DOF with MPU6050 HMC5883L MS5611, LLC`
- `//#define GY_88 // Chinese 10 DOF with MPU6050 HMC5883L BMP085, LLC`
- `#define GY_521 // Chinese 6 DOF with MPU6050, LLC`
- `//#define INNOVWORKS_10DOF // with ITG3200, BMA180, HMC5883, BMP085 available here http://www.diymulticopter.com`
- `//#define INNOVWORKS_6DOF // with ITG3200, BMA180 available here http://www.diymulticopter.com`
- `//#define MultiWiiMega // MEGA + MPU6050+HMC5883L+MS5611 available here http://www.diymulticopter.com`
- `//#define PROTO_DIY // 10DOF mega board`
- `//#define IOI_MINI_MULTIWII// www.bambucopter.com`
- `//#define Bobs_6DOF_V1 // BobsQuads 6DOF V1 with ITG3200 & BMA180`
- `//#define Bobs_9DOF_V1 // BobsQuads 9DOF V1 with ITG3200, BMA180 & HMC5883L`
- `//#define Bobs_10DOF_BMP_V1 // BobsQuads 10DOF V1 with ITG3200, BMA180, HMC5883L & BMP180 - BMP180 is software compatible with BMP085`
- `//#define FLYDUINO_MPU // MPU6050 Break Out onboard 3.3V reg`

- `//#define CRIUS_AIO_PRO`
- `//#define DESQUARED6DOFV2GO` // DEsquared V2 with ITG3200 only
- `//#define DESQUARED6DOFV4` // DEsquared V4 with MPU6050
- `//#define LADYBIRD`
- `//#define MEGAWAP_V2_STD` // available here:
<http://www.multircshop.com> <- confirmed by Alex
- `//#define MEGAWAP_V2_ADV`
- `//#define HK_MultiWii_SE_V2` // Hobbyking board with MPU6050 + HMC5883L + BMP085
- `//#define HK_MultiWii_328P` // Also labeled "Hobbybro" on the back. ITG3205 + BMA180 + BMP085 + NMC5583L + DSM2 Connector (Spektrum Satellite)
- `//#define RCNet_FC` // RCNet FC with MPU6050 and MS561101BA <http://www.rcnet.com>
- `//#define RCNet_FC_GPS` // RCNet FC with MPU6050 + MS561101BA + HMC5883L + UBLOX GPS <http://www.rcnet.com>
- `//#define FLYDU_ULTRA` // MEGA+10DOF+MT3339 FC
- `//#define DIYFLYING_MAGE_V1` // diyflying 10DOF mega board with MPU6050 + HMC5883L + BMP085 <http://www.indoor-flying.hk>
- `//#define MultiWii_32U4_SE` // Hextronik MultiWii_32U4_SE
- `//#define MultiWii_32U4_SE_no_baro` // Hextronik MultiWii_32U4_SE without the MS561101BA to free flash-memory for other functions
- `//#define Flyduino9DOF` // Flyduino 9DOF IMU MPU6050+HMC5883l
- `//#define Nano_Plane` // Multiwii Plane version with tail-front LSM330 sensor http://www.radiosait.ru/en/page_5324.html
-
- `/*
***** independent sensors

*/`
- `/* leave it commented if you already checked a specific board above */`
- `/* I2C gyroscope */`
- `//#define WMP`
- `//#define ITG3050`
- `//#define ITG3200`
- `//#define MPU3050`
- `//#define L3G4200D`
- `//#define MPU6050` //combo + ACC
- `//#define LSM330` //combo + ACC
-
- `/* I2C accelerometer */`
- `//#define MMA7455`

- `//#define ADXL345`
- `//#define BMA020`
- `//#define BMA180`
- `//#define BMA280`
- `//#define LIS3LV02`
- `//#define LSM303DLx_ACC`
- `//#define MMA8451Q`
-
- `/* I2C barometer */`
- `//#define BMP085`
- `//#define MS561101BA`
-
- `/* I2C magnetometer */`
- `//#define HMC5843`
- `//#define HMC5883`
- `//#define AK8975`
- `//#define MAG3110`
-
- `/* Sonar */ // for visualization purpose currently - no control code behind`
- `//#define SRF02 // use the Devantech SRF i2c sensors`
- `//#define SRF08`
- `//#define SRF10`
- `//#define SRF23`
-
- `/* ADC accelerometer */ // for 5DOF from sparkfun, uses analog PIN`
`A1/A2/A3`
- `//#define ADCACC`
-
- `/* enforce your individual sensor orientation - even overrides board`
`specific defaults */`
- `//#define FORCE_ACC_ORIENTATION(X, Y, Z) {imu.accADC[ROLL]`
`= Y; imu.accADC[PITCH] = -X; imu.accADC[YAW] = Z;}`
- `//#define FORCE_GYRO_ORIENTATION(X, Y, Z)`
`{imu.gyroADC[ROLL] = -Y; imu.gyroADC[PITCH] = X;`
`imu.gyroADC[YAW] = Z;}`
- `//#define FORCE_MAG_ORIENTATION(X, Y, Z)`
`{imu.magADC[ROLL] = X; imu.magADC[PITCH] = Y;`
`imu.magADC[YAW] = Z;}`
-
- `/* Board orientation shift */`
- `/* If you have frame designed only for + mode and you cannot rotate FC`
`phisycally for flying in X mode (or vice versa)`

- * you can use one of these options for virtual sensors rotation by 45 degrees, then set type of multicopter according to flight mode.
- * Check motors order and directions of motors rotation for matching with new front point! Uncomment only one option! */
- `//#define SENSORS_TILT_45DEG_RIGHT // rotate the FRONT 45 degrees clockwise`
- `//#define SENSORS_TILT_45DEG_LEFT // rotate the FRONT 45 degrees counterclockwise`
-
-
- `/*

******/`
- `/*

******/`
- `/* SECTION 2 - COPTER TYPE SPECIFIC OPTIONS
******/`
- `/*

******/`
- `/*

******/`
- `/* PID Controller
******/`
- `/* choose one of the alternate PID control algorithms`
- * 1 = evolved oldschool algorithm (similar to v2.2)
- * 2 = new experimental algorithm from Alex Khoroshko - unsupported -
<http://www.multiwii.com/forum/viewtopic.php?f=8&t=3671&start=10#p3738>
- * */
- `#define PID_CONTROLLER 1`
-
- `/* NEW: not used anymore for servo coptertypes <== NEEDS FIXING -
MOVE TO WIKI */`
- `#define YAW_DIRECTION 1`
- `//#define YAW_DIRECTION -1 // if you want to reverse the yaw
correction direction`
-
- `#define ONLYARMWHENFLAT //prevent the copter from arming when
the copter is tilted`
-
- `/*
***** ARM/DISARM
******/`
- `/* optionally disable stick combinations to arm/disarm the motors.`

- * In most cases one of the two options to arm/disarm via TX stick is sufficient */
- #define ALLOW_ARM_DISARM_VIA_TX_YAW
- // #define ALLOW_ARM_DISARM_VIA_TX_ROLL
-
- /***** SERVOES *****/
- /* info on which servos connect where and how to setup can be found here
- *
http://www.multiwii.com/wiki/index.php?title=Config.h#Servos_configuration
- */
-
- /* Do not move servos if copter is unarmed
- * It is a quick hack to overcome feedback tail wigglight when copter has a flexible
- * landing gear
- */
- // #define DISABLE_SERVOS_WHEN_UNARMED
-
-
- /* if you want to preset min/middle/max values for servos right after flashing, because of limited physical
- * room for servo travel, then you must enable and set all three following options */
- // #define SERVO_MIN { 1020, 1020, 1020, 1020, 1020, 1020, 1020, 1020 }
- // #define SERVO_MAX { 2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000 }
- // #define SERVO_MID { 1500, 1500, 1500, 1500, 1500, 1500, 1500, 1500 } // (*)
- // #define FORCE_SERVO_RATES { 30,30,100,100,100,100,100,100 }
// 0 = normal, 1= reverse
-
- /***** Cam Stabilisation *****/
- /* The following lines apply only for a pitch/roll tilt stabilization system. Uncomment the first or second line to activate it */
- // #define SERVO_MIX_TILT
- // #define SERVO_TILT
-

- /* camera trigger function : activated via Rc Options in the GUI, servo output=A2 on promini */
- // trigger interval can be changed via (*GUI*) or via AUX channel
- //define CAMTRIG
- #define CAM_TIME_HIGH 1000 // the duration of HIGH state servo expressed in ms
-
- /***** Airplane *****/
- //define USE_THROTTLESERVO // For use of standard 50Hz servo on throttle.
-
- //define FLAPPERONS AUX4 // Mix Flaps with Ailerons.
- #define FLAPPERON_EP { 1500, 1700 } // Endpooints for flaps on a 2 way switch else set { 1020,2000 } and program in radio.
- #define FLAPPERON_INVERT { -1, 1 } // Change direction om flapperons { Wing1, Wing2 }
-
- //define FLAPS // Traditional Flaps on SERVO3.
- //define FLAPSPEED 3 // Make flaps move slowm Higher value is Higher Speed.
-
- /***** Common for Heli & Airplane *****/
-
- /* Governor: attempts to maintain rpm through pitch and voltage changes
- * predictive approach: observe input signals and voltage and guess appropriate corrections.
- * (the throttle curve must leave room for the governor, so 0-50-75-80-80 is ok, 0-50-95-100-100 is _not_ ok.
- * Can be toggled via aux switch.
- */
- //define GOVERNOR_P 7 // (*) proportional factor. Higher value -> higher throttle increase. Must be >=1; 0 = turn off
- //define GOVERNOR_D 4 // (*) decay timing. Higher value -> takes longer to return throttle to normal. Must be >=1;
-
- /* tail precomp from collective */
- #define YAW_COLL_PRECOMP 10 // (*) proportional factor in 0.1. Higher value -> higher precomp effect. value of 10 equals no/neutral effect
- #define YAW_COLL_PRECOMP_DEADBAND 120 // (*) deadband for collective pitch input signal around 0-pitch input value

-
- `//#define VOLTAGEDROP_COMPENSATION // voltage impact correction`
-
- `/****** Heli
******/`
- `/* Channel to control CollectivePitch */`
- `#define COLLECTIVE_PITCH THROTTLE`
-
- `/* Limit the range of Collective Pitch. 100% is Full Range each way and position for Zero Pitch */`
- `#define COLLECTIVE_RANGE { 80, 0, 80 } // {Min%, ZeroPitch offset from 1500, Max% }.`
- `#define YAWMOTOR 0 // If a motor is used as YAW Set to 1 else set to 0.`
-
- `/* Servo mixing for heli 120
{ Coll,Nick,Roll } */`
- `#define SERVO_NICK { +10, -10, 0 }`
- `#define SERVO_LEFT { +10, +5, +10 }`
- `#define SERVO_RIGHT { +10, +5, -10 }`
-
- `/* Limit Maximum controll for Roll & Nick in 0-100% */`
- `#define CONTROL_RANGE { 100, 100 } // { ROLL,PITCH }`
-
- `/* use servo code to drive the throttle output. You want this for analog servo driving the throttle on IC engines.`
- `if inactive, throttle output will be treated as a motor output, so it can drive an ESC */`
- `//#define HELI_USE_SERVO_FOR_THROTTLE`
-
- `/****** your individual mixing
******/`
- `/* if you want to override an existing entry in the mixing table, you may want to avoid editing the`
- `* mixTable() function for every version again and again.`
- `* howto:`
- `http://www.multiwii.com/wiki/index.php?title=Config.h#Individual_Mixing`
- `*/`
- `//#define MY_PRIVATE_MIXING "filename.h"`
-

- `/****** your individual defaults
******/`
- `/* if you want to replace the hardcoded default values with your own (e.g.
from a previous save to an .mwi file),`
- `* you may want to avoid editing the LoadDefaults() function for every
version again and again.`
- `*`
- `http://www.multiwii.com/wiki/index.php?title=Config.h#Individual_defaults`
- `*/`
- `//#define MY_PRIVATE_DEFAULTS "filename.h"`
-
-
- `/******
******/`
- `/******
******/`
- `/****** SECTION 3 - RC SYSTEM SETUP
******/`
- `/******
******/`
- `/******
******/`
- `/* note: no need to uncomment something in this section if you use a
standard receiver */`
-
- `/****** EXTENDED AUX STATES
******/`
- `/* If you uncomment this line, you can use six states for each of the aux
channels (AUX1-AUX4)`
- `to control your copter.`
- `Channel values`
- `1000-1230`
- `1231-1360`
- `1361-1490`
- `1491-1620`
- `1621-1749`
- `1750-`
-
- `At this moment you can use this function only with WinGUI 2.3 release.
MultiWiiConf does not support it yet`
- `*/`

-
- `//#define EXTENDED_AUX_STATES`
-
-
-
- `/*

******/`
- `/*
***** special receiver types
******/`
-
- `/*

******/`
-
- `/*
***** PPM Sum Reciver
******/`
- `/* The following lines apply only for specific receiver with only one PPM
sum signal, on digital PIN 2`
- `Select the right line depending on your radio brand. Feel free to modify
the order in your PPM order is different */`
- `//#define SERIAL_SUM_PPM
PITCH,YAW,THROTTLE,ROLL,AUX1,AUX2,AUX3,AUX4,8,9,10,11
//For Graupner/Spektrum`
- `//#define SERIAL_SUM_PPM
ROLL,PITCH,THROTTLE,YAW,AUX1,AUX2,AUX3,AUX4,8,9,10,11
//For Robe/Hitec/Futaba`
- `//#define SERIAL_SUM_PPM
ROLL,PITCH,YAW,THROTTLE,AUX1,AUX2,AUX3,AUX4,8,9,10,11
//For Multiplex`
- `//#define SERIAL_SUM_PPM
PITCH,ROLL,THROTTLE,YAW,AUX1,AUX2,AUX3,AUX4,8,9,10,11
//For some Hitec/Sanwa/Others`
-
- `// Uncommenting following line allow to connect PPM_SUM receiver to
standard THROTTLE PIN on MEGA boards (eg. A8 in CRIUS AIO)`
- `//#define PPM_ON_THROTTLE`
-
- `/*
***** Spektrum Satellite Reciver
******/`
- `/* The following lines apply only for Spektrum Satellite Receiver`
- `Spektrum Satellites are 3V devices. DO NOT connect to 5V!`

- For MEGA boards, attach sat grey wire to RX1, pin 19. Sat black wire to ground. Sat orange wire to Mega board's 3.3V (or any other 3V to 3.3V source).
- For PROMINI, attach sat grey to RX0. Attach sat black to ground. */
- `//#define SPEKTRUM 1024`
- `//#define SPEKTRUM 2048`
- `//#define RX_SERIAL_PORT 1 // Forced to 0 on Pro Mini and single serial boards; Set to your choice of 0, 1, or 2 on any Mega based board (defaults to 1 on Mega).`
- `//*****`
- `// Defines that allow a "Bind" of a Spektrum or Compatible Remote Receiver (aka Satellite) via Configuration GUI.`
- `// Bind mode will be same as declared above, if your TX is capable.`
- `// Ground, Power, and Signal must come from three adjacent pins.`
- `// By default, these are Ground=4, Power=5, Signal=6. These pins are in a row on most MultiWii shield boards. Pins can be overridden below.`
- `// Normally use 3.3V regulator is needed on the power pin!! If your satellite hangs during bind (blinks, but won't complete bind with a solid light), go direct 5V on all pins.`
- `//*****`
- `// For Pro Mini, the connector for the Satellite that resides on the FTDI can be unplugged and moved to these three adjacent pins.`
- `//#define SPEK_BIND //Un-Comment for Spektrum Satellie Bind Support. Code is ~420 bytes smaller without it.`
- `//#define SPEK_BIND_GROUND 4`
- `//#define SPEK_BIND_POWER 5`
- `//#define SPEK_BIND_DATA 6`
- `/****** SBUS RECIVER ******/`
- `/* The following line apply only for Futaba S-Bus Receiver on MEGA boards or PROMICRO boards.`
- `You have to invert the S-Bus-Serial Signal e.g. with a Hex-Inverter like IC SN74 LS 04 */`
- `//#define SBUS`
`PITCH,YAW,THROTTLE,ROLL,AUX1,AUX2,AUX3,AUX4,8,9,10,11,12,13,14,15,16,17 // dsm2 orangerx`
- `//#define SBUS`
`ROLL,PITCH,THROTTLE,YAW,AUX1,AUX2,AUX3,AUX4,8,9,10,11,12,13,14,15,16,17 // T14SG`
- `//#define RX_SERIAL_PORT 1`
- `#define SBUS_MID_OFFSET 988 //SBUS Mid-Point at 1500`

-
- /***** HOTT RECIVER
*****/
- /* Graupner Hott HD */
- //#define SUMD
PITCH,YAW,THROTTLE,ROLL,AUX1,AUX2,AUX3,AUX4
- //#define RX_SERIAL_PORT 1
-
- /*****
*****/
- /*****
*****/
- /***** SECTION 4 - ALTERNATE CPU's & BOARDS
*****/
- /*****
*****/
- /*****
*****/
-
-
- /*****
*****/
- /***** Promini Specifig Settings
*****/
-
- /*****
*****/
-
- /***** Hexa Motor 5 & 6 Pins
*****/
- /* PIN A0 and A1 instead of PIN D5 & D6 for 6 motors config and
promini config
- This mod allow the use of a standard receiver on a pro mini
- (no need to use a PPM sum receiver) */
- //#define A0_A1_PIN_HEX
-
- /***** Aux 2 Pin
*****/
- /* possibility to use PIN8 or PIN12 as the AUX2 RC input (only one, not
both)
- it deactivates in this case the POWER PIN (pin 12) or the BUZZER PIN
(pin 8) */

```

//#define RCAUXPIN8
//#define RCAUXPIN12

/*****
*****/

/***** Teensy 2.0 Support *****/

/*****

/* uncomment this if you use a teensy 2.0 with teensyduino
   it needs to run at 16MHz */
//#define TEENSY20

/*****
*****/

/***** Settings for ProMicro, Leonardo and other Atmega32u4 Boards *****/

/*****
*****/

/***** pin Layout *****/

/* activate this for a better pinlayout if all pins can be used => not possible
on ProMicro */
//#define A32U4ALLPINS

/***** PWM Setup *****/

/* activate all 6 hardware PWM outputs Motor 5 = D11 and 6 = D13.
   note: not possible on the sparkfun promicro (pin 11 & 13 are not broken
   out there)

   if activated:
   Motor 1-6 = 10-bit hardware PWM
   Motor 7-8 = 8-bit Software PWM
   Servos   = 8-bit Software PWM

   if deactivated:

```

- Motor 1-4 = 10-bit hardware PWM
- Motor 5-8 = 10-bit Software PWM
- Servos = 10-bit Software PWM */
- `//#define HWPWM6`
-
- `/* ***** Aux 2 Pin ***** */`
- `/* AUX2 pin on pin RXO */`
- `//#define RCAUX2PINRXO`
-
- `/* aux2 pin on pin D17 (RXLED) */`
- `//#define RCAUX2PIND17`
-
- `/* ***** Buzzer Pin ***** */`
- `/* this moves the Buzzer pin from TXO to D8 for use with ppm sum or spectrum sat. RX (not needed if A32U4ALLPINS is active) */`
- `#define D8BUZZER`
-
- `/* ***** Promicro version related ***** */`
- `/* Inverted status LED for Promicro ver 10 */`
- `//#define PROMICRO10`
-
-
-
- `/* ***** override default pin assignments ***** */`
-
- `/* ***** */`
-
- `/* only enable any of this if you must change the default pin assignment, e.g. your board does not have a specific pin */`
- `/* you may need to change PINx and PORTx plus #shift according to the desired pin! */`
- `//#define OVERRIDE_V_BATPIN A0 // instead of A3 //`
- Analog PIN 3
-

- `//#define OVERRIDE_PSENSORPIN A1 // instead of A2 //`
Analog PIN 2
-
- `//#define OVERRIDE_LEDPIN_PINMODE pinMode (A1,`
`OUTPUT); // use A1 instead of d13`
- `//#define OVERRIDE_LEDPIN_TOGGLE PINC |= 1<<1; // PINB`
`|= 1<<5; //switch LEDPIN state (digital PIN 13)`
- `//#define OVERRIDE_LEDPIN_OFF PORTC &= ~(1<<1); //`
`PORTB &= ~(1<<5);`
- `//#define OVERRIDE_LEDPIN_ON PORTC |= 1<<1; // was`
`PORTB |= (1<<5);`
-
- `//#define OVERRIDE_BUZZERPIN_PINMODE pinMode (A2,`
`OUTPUT); // use A2 instead of d8`
- `//#define OVERRIDE_BUZZERPIN_ON PORTC |= 1<<2`
`//PORTB |= 1;`
- `//#define OVERRIDE_BUZZERPIN_OFF PORTC &= ~(1<<2);`
`//PORTB &= ~1;`
-
- `/*

*/`
- `/*

*/`
- `/* ***** SECTION 5 - ALTERNATE SETUP

*/`
- `/*

*/`
- `/*

*/`
-
- `/* ***** Serial com speed

*/`
- `/* This is the speed of the serial interfaces */`
- `#define SERIAL0_COM_SPEED 115200`
- `#define SERIAL1_COM_SPEED 115200`
- `#define SERIAL2_COM_SPEED 115200`
- `#define SERIAL3_COM_SPEED 115200`
-
- `/* when there is an error on I2C bus, we neutralize the values during a short`
time. expressed in microseconds
- `it is relevent only for a conf with at least a WMP */`
- `#define NEUTRALIZE_DELAY 100000`

- ```

/***** Moving Average Gyros
*****/

```
- ```

#define MMGYRO 10 // (*) Active Moving Average
Function for Gyros

```
- ```

#define MMGYROVECTORLENGTH 15 // Length of Moving
Average Vector (maximum value for tunable MMGYRO

```
- ```

/* Moving Average ServoGimbal Signal Output */

```
- ```

#define MMSERVOGIMBAL // Active Output Moving
Average Function for Servos Gimbal

```
- ```

#define MMSERVOGIMBALVECTORLENGTH 32 // Length of
Moving Average Vector

```
-
- ```

/***** Analog Reads
*****/

```
- ```

/* if you want faster analog Reads, enable this. It may result in less accurate
results, especially for more than one analog channel */

```
- ```

#define FASTER_ANALOG_READS

```
- 
- ```

/*****
*****/

```
- ```

/*****
*****/

```
- ```

/***** SECTION 6 - OPTIONAL FEATURES
*****/

```
- ```

/*****
*****/

```
- ```

/*****
*****/

```
-
- ```

/***** Reset Baro altitude on arm
*****/

```
- ```

/* When unchecked a calibration of the baro altitude is preformed every time
arming is activated */

```
- ```

#define ALTITUDE_RESET_ON_ARM

```
- 
- ```

/***** Angele throttle correction
*****/

```
- ```

/* Automatically increase throttle based on the angle of the copter
Original idea by Kraut Rob, first implementation HAdrian */

```
- 
- ```

#define THROTTLE_ANGLE_CORRECTION 40

```
-

- `/** HEADFREE : the copter can be controlled by an absolute stick orientation, whatever the yaw orientation */`
- `//#define HEADFREE`
-
- `/* ***** Advanced Headfree Mode ***** */`
- `/* In Advanced Headfree mode when the copter is farther than ADV_HEADFREE_RANGE meters then`
- `the bearing between home and copter position will become the control direction`
- `IF copter come closer than ADV_HEADFREE_RANGE meters, then the control direction freezed to the`
- `bearing between home and copter at the point where it crosses the ADV_HEADFREE_RANGE meter distance`
- `first implementation by HAdrian, mods by EOSBandi`
- `*/`
-
- `//#define ADVANCED_HEADFREE //Advanced headfree mode is enabled when this is uncommented`
- `//#define ADV_HEADFREE_RANGE 15 //Range where advanced headfree mode activated`
-
-
- `/* ***** continuous gyro calibration ***** */`
- `/* Gyrocalibration will be repeated if copter is moving during calibration. */`
- `//#define GYROCALIBRATIONFAILSAFE`
-
- `/* ***** AP FlightMode ***** */`
- `/** FUNCTIONALITY TEMPORARY REMOVED */`
- `/* Temporarily Disables GPS_HOLD_MODE to be able to make it possible to adjust the Hold-position when moving the sticks.*/`
- `//#define AP_MODE 40 // Create a deadspan for GPS.`
-
- `/* ***** Assisted AcroTrainer ***** */`
- `/* Train Acro with auto recovery. Value set the point where ANGLE_MODE takes over.`
- `Remember to activate ANGLE_MODE first!...`
- `A Value on 200 will give a very distinct transfer */`

- `//#define ACROTRAINER_MODE 200 //`
<http://www.multiwii.com/forum/viewtopic.php?f=16&t=1944#p17437>
-
-
- `/*****` Failsafe settings
`*****/`
- `/*` Failsafe check pulses on four main control channels CH1-CH4. If the pulse is missing or bellow 985us (on any of these four channels)
- the failsafe procedure is initiated. After FAILSAFE_DELAY time from failsafe detection, the level mode is on (if ACC is available),
- PITCH, ROLL and YAW is centered and THROTTLE is set to FAILSAFE_THROTTLE value. You must set this value to descending about 1m/s or so
- for best results. This value is depended from your configuration, AUW and some other params. Next, after FAILSAFE_OFF_DELAY the copter is disarmed,
- and motors is stopped. If RC pulse coming back before reached FAILSAFE_OFF_DELAY time, after the small quard time the RC control is returned to normal. `*/`
- `//#define FAILSAFE` `// uncomment to activate the`
failsafe function
- `#define FAILSAFE_DELAY 10` `// Guard time for failsafe`
activation after signal lost. 1 step = 0.1sec - 1sec in example
- `#define FAILSAFE_OFF_DELAY 200` `// Time for Landing`
before motors stop in 0.1sec. 1 step = 0.1sec - 20sec in example
- `#define FAILSAFE_THROTTLE (MINTHROTTLE + 200)` `// (*)`
Throttle level used for landing - may be relative to MINTHROTTLE - as in this case
-
- `#define FAILSAFE_DETECT_TRESHOLD 985`
-
-
- `/*****` DFRobot LED RING
`*****/`
- `/*` I2C DFRobot LED RING communication `*/`
- `//#define LED_RING`
-
- `/*****` LED FLASHER
`*****/`
- `//#define LED_FLASHER`
- `//#define LED_FLASHER_DDR DDRB`
- `//#define LED_FLASHER_PORT PORTB`

- `//#define LED_FLASHER_BIT PORTB4`
- `//#define LED_FLASHER_INVERT`
- `//#define LED_FLASHER_SEQUENCE 0b00000000 // leds OFF`
- `//#define LED_FLASHER_SEQUENCE_ARMED 0b00000101 //`
create double flashes
- `//#define LED_FLASHER_SEQUENCE_MAX 0b11111111 // full`
illumination
- `//#define LED_FLASHER_SEQUENCE_LOW 0b00000000 // no`
illumination
-
-
- `/* ***** Landing lights`
`***** */`
- `/* Landing lights`
- Use an output pin to control landing lights.
- They can be switched automatically when used in conjunction
- with altitude data from a sonar unit. */
- `//#define LANDING_LIGHTS_DDR DDRC`
- `//#define LANDING_LIGHTS_PORT PORTC`
- `//#define LANDING_LIGHTS_BIT PORTC0`
- `//#define LANDING_LIGHTS_INVERT`
-
- `/* altitude above ground (in cm) as reported by sonar */`
- `//#define LANDING_LIGHTS_AUTO_ALTITUDE 50`
-
- `/* adopt the flasher pattern for landing light LEDs */`
- `//#define LANDING_LIGHTS_ADOPT_LED_FLASHER_PATTERN`
-
- `/* ***** INFLIGHT ACC Calibration`
`***** */`
- `/* This will activate the ACC-Inflight calibration if unchecked */`
- `//#define INFLIGHT_ACC_CALIBRATION`
-
- `/* ***** OSD Switch`
`***** */`
- `// This adds a box that can be interpreted by OSD in activation status (to`
switch on/off the overlay for instance)
- `//#define OSD_SWITCH`
-
-
- `/* *****`
`***** */`

- `/****** TX-related ******/`
- `/******`
- `/* introduce a deadband around the stick center`
- `Must be greater than zero, comment if you dont want a deadband on roll, pitch and yaw */`
- `//#define DEADBAND 6`
- `/******`
- `/******`
- `/****** GPS *****`
- `/******`
- `/******`
- `/* ENable this for using GPS simulator (NMEA only)*/`
- `//#define GPS_SIMULATOR`
- `/* GPS using a SERIAL port`
- `if enabled, define here the Arduino Serial port number and the UART speed`
- `note: only the RX PIN is used in case of NMEA mode, the GPS is not configured by multiwii`
- `in NMEA mode the GPS must be configured to output GGA and RMC NMEA sentences (which is generally the default conf for most GPS devices)`
- `at least 5Hz update rate. uncomment the first line to select the GPS serial port of the arduino */`
- `//#define GPS_SERIAL 2 // should be 2 for flyduino v2. It's the serial port number on arduino MEGA`
- `// must be 0 for PRO_MINI (ex GPS_PRO_MINI)`
- `// note: Now a GPS can share MSP on the same port.`
- `The only constrain is to not use it simultaneously, and use the same port speed.`
- `// avoid using 115200 baud because with 16MHz arduino the 115200 baudrate have more than 2% speed error (57600 have 0.8% error)`

- `#define GPS_BAUD 57600 // GPS_BAUD will override SERIALx_COM_SPEED for the selected port`
-
- `/* GPS protocol`
- `NMEA - Standard NMEA protocol GGA, GSA and RMC sentences are needed`
- `UBLOX - U-Blox binary protocol, use the ublox config file (u-blox-config.ublox.txt) from the source tree`
- `MTK_BINARY16 and MTK_BINARY19 - MTK3329 chipset based GPS with DIYDrones binary firmware (v1.6 or v1.9)`
- `With UBLOX and MTK_BINARY you don't have to use GPS_FILTERING in multiwii code !!! */`
-
-
- `//#define NMEA`
- `//#define UBLOX`
- `//#define MTK_BINARY16`
- `//#define MTK_BINARY19`
- `//#define INIT_MTK_GPS // initialize MTK GPS for using selected speed, 5Hz update rate and GGA & RMC sentence or binary settings`
-
-
- `/* I2C GPS device made with an independant arduino + GPS device including some navigation functions contribution from EOSBandi http://code.google.com/p/i2c-gps-nav/ You have to use at least I2CGpsNav code r33 */`
- `/* all fonctionnalities allowed by SERIAL_GPS are now available for I2C_GPS: all relevant navigation computations are gathered in the main FC */`
-
- `//#define I2C_GPS`
-
- `// If your I2C GPS board has Sonar support enabled`
- `//#define I2C_GPS_SONAR`
-
- `/* indicate a valid GPS fix with at least 5 satellites by flashing the LED - Modified by MIS - Using stable LED (YELLOW on CRIUS AIO) led work as sat number indicator`
- `- No GPS FIX -> LED blink at speed of incoming GPS frames`
- `- Fix and sat no. bellow 5 -> LED off`
- `- Fix and sat no. >= 5 -> LED blinks, one blink for 5 sat, two blinks for 6 sat, three for 7 ... */`
- `#define GPS_LED_INDICATOR`

-
- `//Enables the MSP_WP command set , which is used by WinGUI for displaying an setting up navigation`
- `//#define USE_MSP_WP`
-
- `// HOME position is reset at every arm, uncomment it to prohibit it (you can set home position with GyroCalibration)`
- `//#define DONT_RESET_HOME_AT_ARM`
-
- `/* GPS navigation can control the heading */`
-
- `// copter faces toward the navigation point, maghold must be enabled for it`
- `#define NAV_CONTROLS_HEADING 1 /**)`
- `// true - copter comes in with tail first`
- `#define NAV_TAIL_FIRST 0 /**)`
- `// true - when copter arrives to home position it rotates it's head to takeoff direction`
- `#define NAV_SET_TAKEOFF_HEADING 1 /**)`
-
- `/* Get your magnetic declination from here : http://magnetic-declination.com/`
- `Convert the degree+minutes into decimal degree by ==>`
`degree+minutes*(1/60)`
- `Note the sign on declination it could be negative or positive (WEST or EAST)`
- `Also note, that magnetic declination changes with time, so recheck your value every 3-6 months */`
- `#define MAG_DECLINATION 4.02f /**)`
-
- `// Adds a forward predictive filterig to compensate gps lag. Code based on Jason Short's lead filter implementation`
- `#define GPS_LEAD_FILTER /**)`
-
- `// add a 5 element moving average filter to GPS coordinates, helps eliminate gps noise but adds latency comment out to disable`
- `// use it with NMEA gps only`
- `//#define GPS_FILTERING /**)`
-
- `// if we are within this distance to a waypoint then we consider it reached (distance is in cm)`
- `#define GPS_WP_RADIUS 100 /**)`
-
- `// Safe WP distance, do not start mission if the first wp distance is larger than this number (in meters)`

- // Also aborts mission if the next waypoint distance is more than this number
- #define SAFE_WP_DISTANCE 500 //(**)
-
- //Maximum allowable navigation altitude (in meters) automatic altitude control will not go above this height
- #define MAX_NAV_ALTITUDE 100 //(**)
-
- // minimum speed when approach waypoint
- #define NAV_SPEED_MIN 100 // cm/sec //(**)
- // maximum speed to reach between waypoints
- #define NAV_SPEED_MAX 400 // cm/sec //(**)
- // Slow down to zero when reaching waypoint (same as NAV_SPEED_MIN = 0)
- #define NAV_SLOW_NAV 0 //(**)
- // Weight factor of the crosstrack error in navigation calculations (do not touch)
- #define CROSSTRACK_GAIN .4 //(**)
- // Maximum allowable banking than navigation outputs
- #define NAV_BANK_MAX 3000 //(**)
-
- //Defines the RTH altitude. 0 means keep current alt during RTH (in meters)
- #define RTH_ALTITUDE 15 //(**)
- //Wait to reach RTH alt before start moving to home (0-no, 1-yes)
- #define WAIT_FOR_RTH_ALT 1 //(**)
-
- //Navigation engine will takeover BARO mode control
- #define NAV_TAKEOVER_BARO 1 //(**)
-
- //Throttle stick input will be ignored (only in BARO)
- #define IGNORE_THROTTLE 1 //(**)
-
- //If FENCE DISTANCE is larger than 0 then copter will switch to RTH when it farther from home
- //than the defined number in meters
- #define FENCE_DISTANCE 600
-
- //This governs the descent speed during landing. 100 is equals approx 50cm/sec
- #define LAND_SPEED 100
-
-

- `//#define ONLY_ALLOW_ARM_WITH_GPS_3DFIX // Only allow FC arming if GPS has a 3D fix.`
-
-
- `/*

******/`
- `/* LCD/OLED - display settings
******/`
-
- `/*

******/`
-
- `/*
http://www.multiwii.com/wiki/index.php?title=Extra_features#LCD_.2F_OLED */`
-
- `/* ***** The type of LCD
***** */`
- `/* choice of LCD attached for configuration and telemetry, see notes below */`
- `//#define LCD_DUMMY // No Physical LCD attached. With this & LCD_CONF defined, TX sticks still work to set gains, by watching LED blink.`
- `//#define LCD_SERIAL3W // Alex' initial variant with 3 wires, using rx-pin for transmission @9600 baud fixed`
- `//#define LCD_TEXTSTAR // SERIAL LCD: Cat's Whisker LCD_TEXTSTAR Module CW-LCD-02 (Which has 4 input keys for selecting menus)`
- `//#define LCD_VT100 // SERIAL LCD: vt100 compatible terminal emulation (blueterm, putty, etc.)`
- `//#define LCD_TTY // SERIAL LCD: useful to tweak parameters over cable with arduino IDE 'serial monitor'`
- `//#define LCD_ETPP // I2C LCD: Eagle Tree Power Panel LCD, which is i2c (not serial)`
- `//#define LCD_LCD03 // I2C LCD: LCD03, which is i2c`
- `//#define LCD_LCD03S // SERIAL LCD: LCD03 whit serial 9600 baud comunication enabled.`
- `//#define OLED_I2C_128x64 // I2C LCD: OLED
http://www.multiwii.com/forum/viewtopic.php?f=7&t=1350`
- `//#define OLED_DIGOLE // I2C OLED from
http://www.digole.com/index.php?productID=550`
-

- ```

/***** Display settings
*****/
#define LCD_SERIAL_PORT 0 // must be 0 on Pro Mini and single
serial boards; Set to your choice on any Mega based board

//define SUPPRESS_OLED_I2C_128x64LOGO // suppress display of
OLED logo to save memory

/* double font height for better readability. Reduces visible #lines by half.
* The lower part of each page is accessible under the name of shifted
keyboard letter :
* 1 - ! , 2 - @ , 3 - # , 4 - $, 5 - % , 6 - ^ , 7 - & , 8 - * , 9 - (
* You must add both to your lcd.telemetry.* sequences
*/
#define DISPLAY_FONT_DSIZE //currently only aplicable for
OLED_I2C_128x64 and OLED_DIGOLE

/* style of display - AUTODETECTED via LCD_ setting - only activate to
override defaults */
#define DISPLAY_2LINES
#define DISPLAY_MULTILINE
#define MULTILINE_PRE 2 // multiline configMenu # pref lines
#define MULTILINE_POST 6 // multiline configMenu # post lines
#define DISPLAY_COLUMNS 16
/***** Navigation
*****/
/* keys to navigate the LCD menu */
#define LCD_MENU_PREV 'p'
#define LCD_MENU_NEXT 'n'
#define LCD_VALUE_UP 'u'
#define LCD_VALUE_DOWN 'd'

#define LCD_MENU_SAVE_EXIT 's'
#define LCD_MENU_ABORT 'x'

/*****
*****/
/***** LCD configuration menu
*****/

```

- ```

/*****
*****

```
-
- /* uncomment this line if you plan to use a LCD or OLED for tweaking parameters
- *
http://www.multiwii.com/wiki/index.php?title=Extra_features#Configuration_Menu */
- //#define LCD_CONF
-
- /* to include setting the aux switches for AUX1 -> AUX4 via LCD */
- //#define LCD_CONF_AUX
-
- /* optional exclude some functionality - uncomment to suppress unwanted aux channel configuration options */
- //#define SUPPRESS_LCD_CONF_AUX2
- //#define SUPPRESS_LCD_CONF_AUX34
-
-
- ```

/*****

```
- /\*\*\*\*\*                                   LCD           telemetry  

```

```
- ```

/*****
*****

```
-
- /* to monitor system values (battery level, loop time etc. with LCD
- * http://www.multiwii.com/wiki/index.php?title=LCD_Telemetry */
-
- /***** Activation

```

*****

```
- //#define LCD_TELEMETRY
-
- /* to enable automatic hopping between a choice of telemetry pages
uncomment this. */
- //#define LCD_TELEMETRY_AUTO "123452679" // pages 1 to 9 in
ascending order
- //#define LCD_TELEMETRY_AUTO "212232425262729" // strong
emphasis on page 2
-

-
- `/*
*****/
#define BUZZER
#define RCOPTIONSBEEP // uncomment this if you want the buzzer
to beep at any rcOptions change on channel Aux1 to Aux4
#define ARMEDTIMEWARNING 330 // (*) Trigger an alarm after a
certain time of being armed [s] to save you lipo (if your TX does not have a
countdown)
//#define PILOTLAMP //Uncomment if you are using a X-Aircraft
Pilot Lamp`
-
-
- `/*
*****/
/**** battery voltage monitoring ****/
/*
/*
/*
/* for V BAT monitoring
after the resistor divisor we should get [0V;5V]->[0;1023] on analog
V_BATPIN
with R1=33k and R2=51k
vbat = [0;1023]*16/VBATSCALE
must be associated with #define BUZZER ! */
//#define VBAT // uncomment this line to activate the vbat code
#define VBATSCALE 131 // (*) (**) change this value if readed
Battery voltage is different than real voltage
#define VBATNOMINAL 126 // 12,6V full battery nominal voltage -
only used for lcd.telemetry
#define VBATLEVEL_WARN1 107 // (*) (**) 10,7V
#define VBATLEVEL_WARN2 99 // (*) (**) 9.9V
#define VBATLEVEL_CRIT 93 // (*) (**) 9.3V - critical condition: if
vbat ever goes below this value, permanent alarm is triggered
#define NO_VBAT 16 // Avoid beeping without any battery
#define VBAT_OFFSET 0 // offset in 0.1Volts, gets added to voltage
value - useful for zener diodes`
-
- `/* for V BAT monitoring of individual cells
* enable both VBAT and VBAT_CELLS
*/
//#define VBAT_CELLS`

- ```

 #define VBAT_CELLS_NUM 0 // set this to the number of cells you
monitor via analog pins
• #define VBAT_CELLS_PINS {A0, A1, A2, A3, A4, A5 } // set this to the
sequence of analog pins
• #define VBAT_CELLS_OFFSETS {0, 50, 83, 121, 149, 177 } // in 0.1
volts, gets added to voltage value - useful for zener diodes
• #define VBAT_CELLS_DIVS { 75, 122, 98, 18, 30, 37 } // divisor for
proportional part according to resistors - larger value here gives smaller
voltage
•
•
•
/*****
*****/
• /**** powermeter (battery capacity monitoring) *****/
•
/*****
*****/
•
•
• /* enable monitoring of the power consumption from battery (think of mAh)
allows to set alarm value in GUI or via LCD
• Full description and howto here
http://www.mutiwii.com/wiki/index.php?title=Powermeter
• Two options:
• 1 - hard: - (uses hardware sensor, after configuration gives very good
results)
• 2 - soft: - (good results +-5% for plush and mystery ESCs @ 2S and 3S,
not good with SuperSimple ESC) */
• //#define POWERMETER_SOFT
• //#define POWERMETER_HARD
• #define PSENSORNULL 510 /* (*) hard only: set to analogRead() value
for zero current; for I=0A my sensor
• gives 1/2 Vss; that is approx 2.49Volt; */
• #define PINT2mA 132 /* (*) hard: one integer step on arduino analog
translates to mA (example 4.9 / 37 * 1000) ;
• soft: use fictional value, start with 100.
• for hard and soft: larger PINT2mA will get you larger
value for power (mAh equivalent) */
• //#define WATTS // compute and display the actual watts (=Volt*Ampere)
consumed - requires both POWERMETER_HARD and VBAT
•

```



- 
- `/*  
*****/  
**** altitude hold ****/  
*/`
- 
- `/*  
*****/  
*/`
- 
- `/* defines the neutral zone of throttle stick during altitude hold, default  
setting is`
- `+/-50 uncommend and change the value below if you want to change it. */`
- `#define ALT_HOLD_THROTTLE_NEUTRAL_ZONE 50`
- `//#define ALT_HOLD_THROTTLE_MIDPOINT 1500 // in us - if  
uncommented, this value is used in ALT_HOLD for throttle stick middle point  
instead of initialThrottleHold parameter.`
- 
- 
- `/* uncomment to disable the altitude hold feature.`
- `* This is useful if all of the following apply`
- `* + you have a baro`
- `* + want altitude readout and/or variometer`
- `* + do not use altitude hold feature`
- `* + want to save memory space */`
- `//#define SUPPRESS_BARO_ALTHOLD`
- 
- 
- `/*  
*****/  
**** altitude variometer ****/  
*/`
- 
- `/*  
*****/  
*/`
- 
- `/* enable to get audio feedback upon rising/falling copter/plane.`
- `* Requires a working baro.`
- `* For now, Output gets sent to an enabled vt100 terminal program over the  
serial line.`
- `* choice of two methods (enable either one or both)`
- `* method 1 : use short term movement from baro ( bigger code size)`
- `* method 2 : use long term observation of altitude from baro (smaller code  
size)`
- `*/`

- `//#define VARIOMETER 12 // possible values: 12 = methods 1 & 2 ;`  
`1 = method 1 ; 2 = method 2`
- `//#define SUPPRESS_VARIOMETER_UP // if no signaling for up`  
`movement is desired`
- `//#define SUPPRESS_VARIOMETER_DOWN // if no signaling for down`  
`movement is desired`
- `//#define VARIOMETER_SINGLE_TONE // use only one tone (BEL);`  
`neccessary for non-patched vt100 terminals`
- 
- 
- `/*`  
`*****/`
- `/* board naming */`
- 
- `/*`  
`*****/`
- 
- `/*`
- `* this name is displayed together with the MultiWii version number`
- `* upon powerup on the LCD.`
- `* If you are without a DISPLAYD then You may enable LCD_TTY and`
- `* use arduino IDE's serial monitor to view the info.`
- `*`
- `* You must preserve the format of this string!`
- `* It must be 16 characters total,`
- `* The last 4 characters will be overwritten with the version number.`
- `*/`
- `#define BOARD_NAME "MultiWii V-.-"`
- `// 123456789.123456`
- 
- `/* Support multiple configuration profiles in EEPROM`  
`*****/`
- `//#define MULTIPLE_CONFIGURATION_PROFILES`
- 
- `/* do no reset constants when change of flashed program is`  
`detected *****/`
- `#define NO_FLASH_CHECK`
- 
- `/*`  
`*****/`
- `/*`  
`*****/`

- `/****** SECTION 7 - TUNING & DEVELOPER`  
`*****/`
- `/******`  
`*****/`
- `/******`  
`*****/`
- 
- `#define VBAT_PRESCALER 16 // set this to 8 if vbatscale would exceed`  
`255`
- 
- 
- `/******`  
`*****/`
- `/****** special ESC with extended range [0-2000] microseconds`  
`*****/`
- 
- `/******`  
`*****/`
- `//#define EXT_MOTOR_RANGE // using this with wii-esc requires to`  
`change MINCOMMAND to 1008 for promini and mega`
- 
- 
- `/******`  
`*****/`
- `/****** brushed ESC`  
`*****`  
`*/`
- 
- `/******`  
`*****/`
- `// for 328p proc`
- `//#define EXT_MOTOR_32KHZ`
- `//#define EXT_MOTOR_4KHZ`
- `//#define EXT_MOTOR_1KHZ`
- 
- `// for 32u4 proc`
- `//#define EXT_MOTOR_64KHZ`
- `//#define EXT_MOTOR_32KHZ`
- `//#define EXT_MOTOR_16KHZ`
- `//#define EXT_MOTOR_8KHZ`
-

- ```

/*****
*****

```
- ```

/***** motor, servo and other presets

```
- ```

/*****
*****

```
- ```

/* motors will not spin when the throttle command is in low position
this is an alternative method to stop immediately the motors */

```
- ```

//#define MOTOR_STOP

```
-
- ```

/* some radios have not a neutral point centered on 1500. can be changed
here */

```
- ```

#define MIDRC 1500

```
-
- ```

/***** Servo Refreshrates

```
- ```

/* Default 50Hz Servo refresh rate*/

```
- ```

#define SERVO_RFR_50HZ

```
- 
- ```

/* up to 160Hz servo refreshrate .. works with the most analog servos*/

```
- ```

//#define SERVO_RFR_160HZ

```
- 
- ```

/* up to 300Hz refreshrate it is as fast as possible (100-300Hz depending on
the count of used servos and the servos state).
for use with digital servos
dont use it with analog servos! they may get damage. (some will work but
be careful) */

```
- ```

//#define SERVO_RFR_300HZ

```
- 
- ```

/*****      HW PWM Servos
*****

```
- ```

/* HW PWM Servo outputs for Arduino Mega.. moves:
Pitch = pin 44
Roll = pin 45
CamTrig = pin 46
SERVO4 = pin 11 (aileron left for fixed wing or TRI YAW SERVO)
SERVO5 = pin 12 (aileron right for fixed wing)
SERVO6 = pin 6 (rudder for fixed wing)
SERVO7 = pin 7 (elevator for fixed wing)
SERVO8 = pin 8 (motor for fixed wing)
*/

```

```
#define MEGA_HW_PWM_SERVOS

/* HW PWM Servo outputs for 32u4 NanoWii, MicroWii etc. - works with
either the variable SERVO_RFR_RATE or
 * one of the 3 fixed servo.refresh.rates *
 * Tested only for heli_120, i.e. 1 motor + 4 servos, moves..
 * motor[0] = motor = pin 6
 * servo[3] = nick servo = pin 11
 * servo[4] = left servo = pin 10
 * servo[5] = yaw servo = pin 5
 * servo[6] = right servo= pin 9
*/
//#define A32U4_4_HW_PWM_SERVOS

#define SERVO_RFR_RATE 50 // In Hz, you can set it from 20 to
400Hz, used only in HW PWM mode for mega and 32u4
//#define SERVO_PIN5_RFR_RATE 200 // separate yaw pwm rate.
// In Hz, you can set it from 20 to 400Hz, used only
in HW PWM mode for 32u4

/*****
*****/

/**** Memory savings *****/

/*****
*****/

/* options to counter the general shortage of both flash and ram memory,
like with leonardo m32u4 and others */

/**** suppress handling of serial commands.***
 * This does _not_ affect handling of RXserial, Spektrum or GPS. Those
will not be affected and still work the same.
 * Enable either one or both of the following options */

/* Remove handling of all commands of the New MultiWii Serial Protocol.
 * This will disable use of the GUI, winGUI, android apps and any other
program that makes use of the MSP.
```

- \* You must find another way (like LCD\_CONF) to tune the parameters or live with the defaults.
- \* If you run a LCD/OLED via i2c or serial/Bluetooth, this is safe to use \*/
- `//#define SUPPRESS_ALL_SERIAL_MSP // saves approx 2700 bytes`
- 
- `/* Remove handling of other serial commands.`
- \* This includes navigating via serial the lcd.configuration menu, lcd.telemetry and permanent.log .
- \* Navigating via stick inputs on tx is not affected and will work the same. \*/
- `//#define SUPPRESS_OTHER_SERIAL_COMMANDS // saves approx 0 to 100 bytes, depending on features enabled`
- 
- `/***** suppress keeping the defaults for initial setup and reset in the code.`
- \* This requires a manual initial setup of the PIDs etc. or load and write from defaults.mwi;
- \* reset in GUI will not work on PIDs
- \*/
- `//#define SUPPRESS_DEFAULTS_FROM_GUI`
- 
- `//#define DISABLE_SETTINGS_TAB // Saves ~400bytes on ProMini`
- 
- 
- `/******`
- `*****/`
- `/***** diagnostics *****/`
- 
- `/******`
- `*****/`
- 
- `/* to log values like max loop time and others to come`
- logging values are visible via LCD config
- set to 1, enable 'R' option to reset values, max current, max altitude
- set to 2, adds min/max cycleTimes
- set to 3, adds additional powerconsumption on a per motor basis (this uses the big array and is a memory hog, if POWERMETER <> PM\_SOFT) \*/
- `//#define LOG_VALUES 1`
- 
- `/* Permanent logging to eeprom - survives (most) upgrades and parameter resets.`
- \* used to track number of flights etc. over lifetime of controller board.
- \* Writes to end of eeprom - should not conflict with stored parameters yet.





- 
- `/*  
/*****  
*****/  
*/`
- 
- `/*` to calibrate all ESCs connected to MWii at the same time (useful to avoid unplugging/re-plugging each ESC)
- Warning: this creates a special version of MultiWii Code
- You cannot fly with this special version. It is only to be used for calibrating ESCs
- Read How To at <http://code.google.com/p/multiwii/wiki/ESCsCalibration>
- `*/`
- `#define ESC_CALIB_LOW MINCOMMAND`
- `#define ESC_CALIB_HIGH 2000`
- `//define ESC_CALIB_CANNOT_FLY // uncomment to activate`
- 
- `/*  
**** internal frequencies ****  
****/  
*/`
- `/*` frequencies for rare cyclic actions in the main loop, depend on cycle time
- time base is main loop cycle time - a value of 6 means to trigger the action every 6th run through the main loop
- example: with cycle time of approx 3ms, do action every 6\*3ms=18ms
- value must be [1; 65535] `*/`
- `#define LCD_TELEMETRY_FREQ 23 // to send telemetry data over serial 23 <=> 60ms <=> 16Hz (only sending interlaced, so 8Hz update rate)`
- `#define LCD_TELEMETRY_AUTO_FREQ 967// to step to next telemetry page 967 <=> 3s`
- `#define PSENSOR_SMOOTH 16 // len of averaging vector for smoothing the PSENSOR readings; should be power of 2; set to 1 to disable`
- `#define VBAT_SMOOTH 16 // len of averaging vector for smoothing the VBAT readings; should be power of 2; set to 1 to disable`
- `#define RSSI_SMOOTH 16 // len of averaging vector for smoothing the RSSI readings; should be power of 2; set to 1 to disable`
- 
- 
- `/*  
*****  
*****/  
*/`
- `/*  
**** Dynamic Motor/Prop Balancing ****  
****/  
*/`
- 
- `/*  
*****  
*****/  
*/`
- `/*  
!!! No Fly Mode !!!  
*/`
- 
- `//define DYNBALANCE // (**) Dynamic balancing controlled from Gui`

```

/*****
*****/

/**** Regression testing ****/

/*****
*****/

/* for development only:
 to allow for easier and reproducible config sets for test compiling,
 different sets of config parameters are kept
 together. This is meant to help detecting compile time errors for various
 features in a coordinated way.

 It is not meant to produce your flying firmware
 To use:
 - do not set any options in config.h,
 - enable with #define COPTERTEST 1, then compile
 - if possible, check for the size
 - repeat with other values of 2, 3, 4 etc.
 */
//#define COPTERTEST 1

/*****
*****/

/*****
*****/

/***** SECTION 8 - DEPRECATED *****/
*****/

/*****
*****/

/*****
*****/

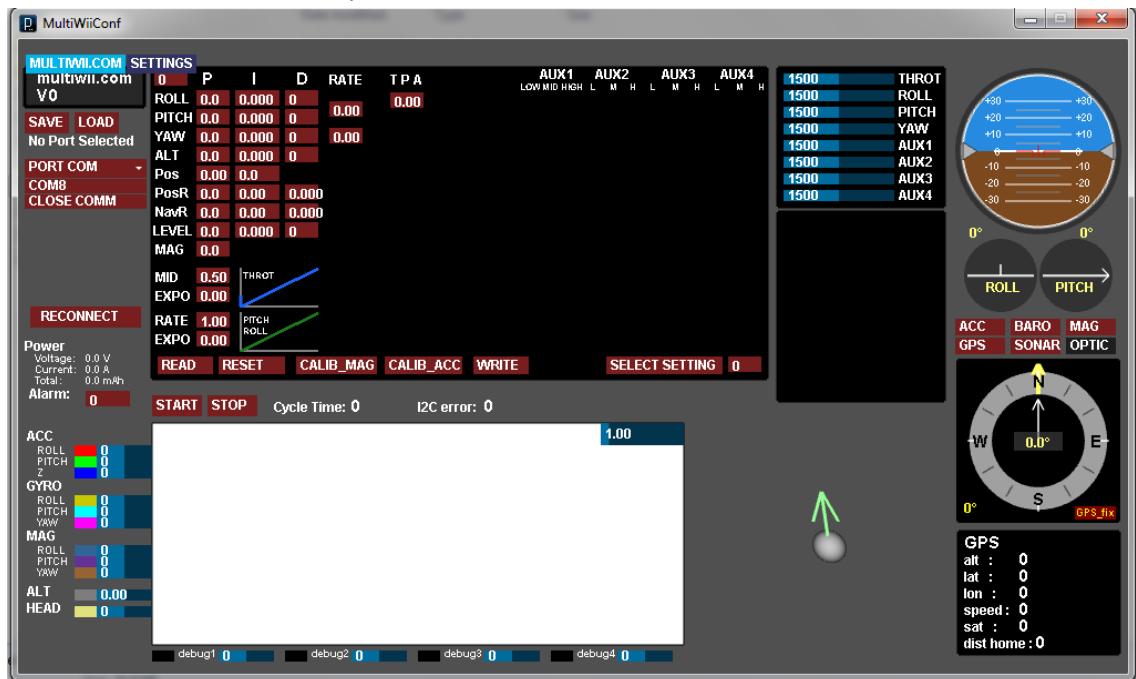
/* these features will be removed in the unforeseeable future. Do not build
new products or
 * functionality based on such features. The default for all such features is
OFF.
*/

/***** WMP power pin *****/
*****/

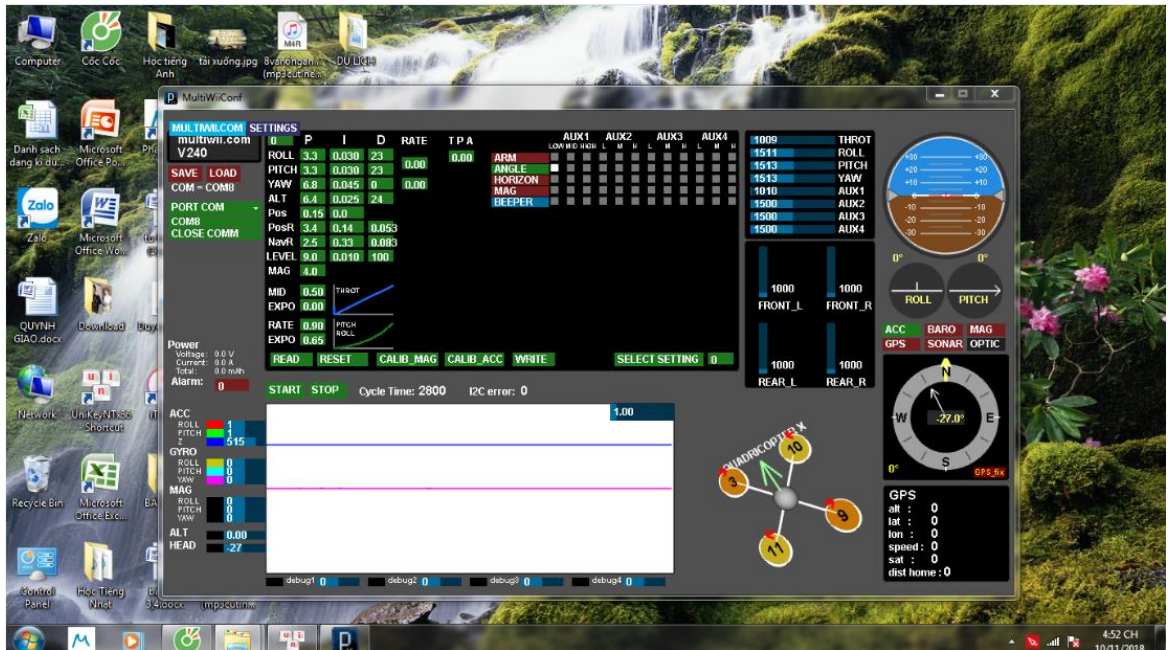
```

- `//#define D12_POWER // Use D12 on PROMINI to power sensors. Will disable servo[4] on D12`
- `/* disable use of the POWER PIN (allready done if the option RCAUXPIN12 is selected) */`
- `#define DISABLE_POWER_PIN`
- `/*  
*****  
*****  
*****/`
- `END OF CONFIGURABLE PARAMETERS`
- `*****/`
- `/*  
*****  
*****  
*****/`
- `#endif /* CONFIG_H_ */`

#### b. Phần mềm hỗ trợ Multiwii 2.4



- Kết nối quadcopter với phần mềm để kiểm tra kênh và Gyro



- Cài đặt cổng com và set up receiver và căn chỉnh PID cho hệ thống
- Set up chế độ mở arm và chế độ bay angle cho phần mềm
- Thực nghiệm bay và căn chỉnh PID phù hợp.

#### IV. Kết luận





