LAMPIRAN A
SPESIFIKASI MOTOR
<table>
<thead>
<tr>
<th>Motor</th>
<th>Propeller</th>
<th>Voltage</th>
<th>Current</th>
<th>Power</th>
<th>Pull</th>
<th>RPM</th>
<th>Pull efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>NX-4005-650kv</td>
<td>GWS 9050</td>
<td>11.5</td>
<td>V</td>
<td>4.45</td>
<td>A</td>
<td>51.18</td>
<td>W</td>
</tr>
<tr>
<td>NX-4005-650kv</td>
<td>GWS 1060</td>
<td>11.5</td>
<td>V</td>
<td>6.4</td>
<td>A</td>
<td>73.60</td>
<td>W</td>
</tr>
<tr>
<td>NX-4005-650kv</td>
<td>GWS 1047</td>
<td>11.5</td>
<td>V</td>
<td>7.8</td>
<td>A</td>
<td>89.70</td>
<td>W</td>
</tr>
<tr>
<td>NX-4005-650kv</td>
<td>GWS 1147</td>
<td>11.5</td>
<td>V</td>
<td>10.46</td>
<td>A</td>
<td>120.29</td>
<td>W</td>
</tr>
<tr>
<td>NX-4005-650kv</td>
<td>GWS 1170</td>
<td>11.5</td>
<td>V</td>
<td>9.13</td>
<td>A</td>
<td>105.00</td>
<td>W</td>
</tr>
<tr>
<td>NX-4005-650kv</td>
<td>GWS 9050</td>
<td>15.3</td>
<td>V</td>
<td>7.63</td>
<td>A</td>
<td>116.74</td>
<td>W</td>
</tr>
<tr>
<td>NX-4005-650kv</td>
<td>GWS 1060</td>
<td>15.3</td>
<td>V</td>
<td>10</td>
<td>A</td>
<td>153.00</td>
<td>W</td>
</tr>
<tr>
<td>NX-4005-650kv</td>
<td>GWS 1047</td>
<td>15.3</td>
<td>V</td>
<td>11.95</td>
<td>A</td>
<td>182.84</td>
<td>W</td>
</tr>
<tr>
<td>NX-4005-650kv</td>
<td>GWS 1147</td>
<td>15.3</td>
<td>V</td>
<td>15</td>
<td>A</td>
<td>229.50</td>
<td>W</td>
</tr>
<tr>
<td>NX-4005-650kv</td>
<td>GWS 1170</td>
<td>15.3</td>
<td>V</td>
<td>13.8</td>
<td>A</td>
<td>211.14</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>GWS 9050</td>
<td>11.5</td>
<td>V</td>
<td>3</td>
<td>A</td>
<td>34.50</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>GWS 1047</td>
<td>11.5</td>
<td>V</td>
<td>5.2</td>
<td>A</td>
<td>59.80</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>GWS 1147</td>
<td>11.5</td>
<td>V</td>
<td>6.85</td>
<td>A</td>
<td>78.78</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>GWS Q1280</td>
<td>11.5</td>
<td>V</td>
<td>8.6</td>
<td>A</td>
<td>98.90</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>GWS 9050</td>
<td>15.3</td>
<td>V</td>
<td>4.7</td>
<td>A</td>
<td>71.91</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>GWS 1047</td>
<td>15.3</td>
<td>V</td>
<td>8</td>
<td>A</td>
<td>122.40</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>APC 12x6E</td>
<td>11.5</td>
<td>V</td>
<td>10.65</td>
<td>A</td>
<td>122.48</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>GWS 1060</td>
<td>15.3</td>
<td>V</td>
<td>6.4</td>
<td>A</td>
<td>97.92</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>GWS 1147</td>
<td>15.3</td>
<td>V</td>
<td>10.5</td>
<td>A</td>
<td>160.65</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>GWS 9050</td>
<td>19.2</td>
<td>V</td>
<td>6.8</td>
<td>A</td>
<td>130.56</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>GWS 1060</td>
<td>19.2</td>
<td>V</td>
<td>9</td>
<td>A</td>
<td>172.80</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>GWS Q1280</td>
<td>15.3</td>
<td>V</td>
<td>13.3</td>
<td>A</td>
<td>203.49</td>
<td>W</td>
</tr>
<tr>
<td>NX-4006-530kv</td>
<td>APC 12x6E</td>
<td>15.3</td>
<td>V</td>
<td>16.35</td>
<td>A</td>
<td>250.16</td>
<td>W</td>
</tr>
<tr>
<td>NX-4008-620kv</td>
<td>GWS 9050</td>
<td>11.5</td>
<td>V</td>
<td>4.5</td>
<td>A</td>
<td>51.75</td>
<td>W</td>
</tr>
<tr>
<td>NX-4008-620kv</td>
<td>GWS 1047</td>
<td>11.5</td>
<td>V</td>
<td>8.3</td>
<td>A</td>
<td>95.45</td>
<td>W</td>
</tr>
<tr>
<td>NX-4008-620kv</td>
<td>GWS 1060</td>
<td>11.5</td>
<td>V</td>
<td>6.5</td>
<td>A</td>
<td>74.75</td>
<td>W</td>
</tr>
<tr>
<td>NX-4008-620kv</td>
<td>10x4.5</td>
<td>11.5</td>
<td>V</td>
<td>8.3</td>
<td>A</td>
<td>95.45</td>
<td>W</td>
</tr>
<tr>
<td>NX-4008-620kv</td>
<td>GWS 1147</td>
<td>11.5</td>
<td>V</td>
<td>10.9</td>
<td>A</td>
<td>125.35</td>
<td>W</td>
</tr>
<tr>
<td>NX-4008-620kv</td>
<td>12.4.5</td>
<td>11.5</td>
<td>V</td>
<td>12.7</td>
<td>A</td>
<td>146.05</td>
<td>W</td>
</tr>
<tr>
<td>NX-4008-620kv</td>
<td>GWS 9050</td>
<td>15</td>
<td>V</td>
<td>7</td>
<td>A</td>
<td>105.00</td>
<td>W</td>
</tr>
<tr>
<td>NX-4008-620kv</td>
<td>GWS 1047</td>
<td>15</td>
<td>V</td>
<td>12.9</td>
<td>A</td>
<td>193.50</td>
<td>W</td>
</tr>
<tr>
<td>NX-4008-620kv</td>
<td>GWS 1060</td>
<td>15</td>
<td>V</td>
<td>10.1</td>
<td>A</td>
<td>151.50</td>
<td>W</td>
</tr>
<tr>
<td>NX-4008-620kv</td>
<td>GWS 1147</td>
<td>15</td>
<td>V</td>
<td>17</td>
<td>A</td>
<td>255.00</td>
<td>W</td>
</tr>
<tr>
<td>NX-4008-620kv</td>
<td>10x4.5</td>
<td>15</td>
<td>V</td>
<td>12.9</td>
<td>A</td>
<td>193.50</td>
<td>W</td>
</tr>
</tbody>
</table>
LAMPIRAN B
PROGRAM MULTIWII
# The type of multicopter

```c
#define GIMBAL
#define BI
#define TRI
#define QUADP
#define QUADX
```
//#define Y4
//#define Y6
//#define HEX6
//#define HEX6X
//#define HEX6H // New Model
//#define OCTO8
//#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
*******************************************/
#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

/**************************************************************************
************
***********
******          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
     // define FFIMUv2     // second version of 9DOF+baro board from Jussi, with HMC5883
     // 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
     // define FREEIMUv035_BMP // FreeIMU v0.3.5_BMP
     // define FREEIMUv04  // FreeIMU v0.4 with MPU6050, HMC5883L, MS5611018A
     // 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
     // define PIPO        // 9DOF board from earrz
     // define QUADRINO    // full FC board 9DOF+baro board from witespy with BMP085 baro
     // 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
     // define ALLINONE    // full FC board or standalone 9DOF+baro board from CS6_EU
     // define AEROQUADSHIELDV2
     // define ATVRSBIN1   // Atmel 9DOF (Contribution by EOSBandi). requires 3.3V power.
     // define SIRIUS      // Sirius Navigator IMU
     // define SIRIUS      // Sirius Navigator IMU

<- confirmed by Alex
#define SIRIUSGPS     // Sirius Navigator IMU using external MAG on GPS board
#define SIRIUS600     // Sirius Navigator IMU using the WMP for the gyro
#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 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
#define GY_85        // Chinese 9 DOF with ITG3205 ADXL345
#define GY_86        // Chinese 10 DOF with MPU6050 HMC5883L MS5611, 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
#define CRIUS_AIO_PRO_V1
#define DESQUARED6DOFV2G0 // DEsquared V2 with ITG3200 only
#define DESQUARED6DOFV4 // DEsquared V4 with MPU6050
#define LADYBIRD
/** independent sensors
*******************************

/* leave it commented if you already checked a specific board above */

*I2C gyroscope* /
#define WMP
#define ITG3200
#define L3G4200D
#define MPU6050 // combo + ACC

*I2C accelerometer* /
#define NUNCHUCK // if you want to use the nunckuck connected to a WMP
#define MMA7455
#define ADXL345
#define BMA020
#define BMA180
#define NUNCHACK // if you want to use the nunckuck as a standalone I2C ACC without WMP
#define LIS3LV02
#define LSM303DLx_ACC

*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

/* individual sensor orientation */
#endif

#endif

#define ACC_ORIENTATION(X, Y, Z)  {accADC[ROLL]  =  Y; accADC[PITCH]  = -X; accADC[YAW]  = Z;}
#define GYRO_ORIENTATION(X, Y, Z) {gyroADC[ROLL] = -Y; gyroADC[PITCH] =  X; gyroADC[YAW] = Z;}
#define MAG_ORIENTATION(X, Y, Z)  {magADC[ROLL]  =  X; magADC[PITCH]  =  Y; magADC[YAW]  = Z;}

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

/**************************** Board orientation shift*************************/
#ifndef
/* If you have frame designed only for + mode and you cannot rotate
   FC physically for flying in X mode (or vice versa)
   you can use one of of this options for virtual sensors rotation by
   45 deegres, then set type of multicopter according to flight mode.
   Check motors order and directions of motors rotation for matching
   with new front point! Uncommment only one option! */

#define SENSORS_TILT_45DEG_RIGHT  // rotate the FRONT 45 deegres
#define SENSORS_TILT_45DEG_LEFT   // rotate the FRONT 45 deegres

#define YAW_DIRECTION 1
#define YAW_DIRECTION -1 // if you want to reverse the yaw correction
direction

/* you can change the tricopter servo travel here */
#define TRI_YAW_CONSTRAINT_MIN 1020
#define TRI_YAW_CONSTRAINT_MAX 2000
#define TRI_YAW_MIDDLE 1500 // (*) tail servo center pos. - use this
for initial trim; later trim midpoint via LCD

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

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

/****************************** 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
#define TILT_PITCH_MIN 1020 //servo travel min, don't set it
below 1020
#define TILT_PITCH_MAX 2000 //servo travel max, max value=2000
#define TILT_PITCH_MIDDLE 1500 //servo neutral value
#define TILT_PITCH_PROP 10 //servo proportional (tied to angle)
; can be negative to invert movement
#define TILT_PITCH_AUX_CH AUX3 //AUX channel to overwrite CAM pitch
(AUX1-AUX4), comment to disable manual input and free the AUX channel
#define TILT_ROLL_MIN 1020
#define TILT_ROLL_MAX 2000
#define TILT_ROLL_MIDDLE 1500
#define TILT_ROLL_PROP 10
#define TILT_ROLL_AUX_CH AUX4 //AUX channel to overwrite CAM Roll
(AUX1-AUX4), comment to disable manual input and free the AUX channel

/* camera trigger function : activated via Rc Options in the GUI, servo
output=A2 on promini */
#define CAMTRIG
#define CAM_SERVO_HIGH 2000 // the position of HIGH state servo
#define CAM_SERVO_LOW 1020 // the position of LOW state servo
#define CAM_TIME_HIGH 1000 // the duration of HIGH state servo
expressed in ms
#define CAM_TIME_LOW 1000 // the duration of LOW state servo
expressed in ms

/****************************** Flying Wing
*******************************/

/* you can change change servo orientation and servo min/max values
here
valid for all flight modes, even passThrough mode
need to setup servo directions here; no need to swap servos amongst
channels at rx */
#define PITCH_DIRECT_L 1 // left servo - pitch orientation
#define PITCH_DIRECT_R -1 // right servo - pitch orientation
(opposite sign to PITCH_DIRECT_L, if servos are mounted in mirrored
orientation)
#define ROLL_DIRECT_L 1 // left servo - roll orientation
#define ROLL_DIRECTION_R 1 // right servo - roll orientation (same sign as ROLL_DIRECTION_L, if servos are mounted in mirrored orientation)
#define WING_LEFT_MID 1500 // (*) left servo center pos. - use this for initial trim; later trim midpoint via LCD
#define WING_RIGHT_MID 1500 // (*) right servo center pos. - use this for initial trim; later trim midpoint via LCD
#define WING_LEFT_MIN 1020 // limit servo travel range must be inside [1020;2000]
#define WING_LEFT_MAX 2000 // limit servo travel range must be inside [1020;2000]
#define WING_RIGHT_MIN 1020 // limit servo travel range must be inside [1020;2000]
#define WING_RIGHT_MAX 2000 // limit servo travel range must be inside [1020;2000]

/******************************
Airplane
******************************

#define USE_THROTTLESERVO // For use of standard 50Hz servo on throttle.
#define SERVO_RATES {100, 100, 100, 100, 100, 100, 100, 100} // Rates in 0-100%
#define SERVO_DIRECTION { -1, 1, 1, -1, 1, 1, 1, 1 } // Invert servos by setting -1

#define FLAPPERONS AUX4 // Mix Flaps with Ailerons.
#define FLAPPERON_EP { 1500, 1700 } // Endpoints for flaps on a 2 way switch else set {1020,2000} and program in radio.
#define FLAPPERON_INVERT { 1, -1 } // Change direction on flapperons { Wing1, Wing2 }

#define FLAPS AUX4 // Traditional Flaps on A2 invert with SERVO_DIRECTION servo[2].
#define FLAP_EP { 1500, 1900 } // Endpoints for flaps on a 2 way switch else set {1020,2000} and program in radio.

#define FLAPSPED 3 // Make flaps move slowm Higher value is Higher Speed.

/******************************
Common for Heli & Airplane
******************************

#if define D12_POWER // Use D12 on PROMINI to power sensors. Will disable servo[4] on D12
#define SERVO_OFFSET { 0, 0, 0, 0, 0, 0, 0 } // (*)
Adjust Servo MID Offset & Swash angles
// Selectable channels:= ROLL,PITCH,THROTTLE,YAW,AUX1,AUX2,AUX3,AUX4

/******************************
Heli
******************************

/* Channel to control CollectivePitch */
#define COLLECTIVE_PITCH THROTTLE

/ * Set Maximum available movement for the servos. Depending on model */
#define SERVO_ENDPOINT_LOW {1020,1020,1020,1020,1020,1020,1020,1020};

/ * Limit the range of Collective Pitch. 100% is Full Range each way and position for Zero Pitch */
#define COLLECTIVE_RANGE { 80, 1500, 80 } // {Min%, ZeroPitch, Max%}.
#define YAWMOTOR 0 // If a motor is used as YAW Set to 1 else set to 0.

/* Servo mixing for heli 120 Use 1/10 fractions (ex.5 = 5/10 = 1/2)
 {Coll,Nick,Roll} */
#define SERVO_NICK { +10, -10, -0 }
#define SERVO_LEFT { +10, +5, +10 }
#define SERVO_RIGHT { +10, +5, -10 }

/ * Servo mixing for heli 90
 {Coll,Nick,Roll} */
#define SERVO_DIRECTIONS { +1, -1, -1 } // -1 will invert servo

/* Use SERVO_OFFSET and SERVO_RATES in Heli and Airplane section for centering and endpoints */

/ *********************** Single and DualCopter Settings
**************************/
/* Change to -1 to reverse servomovement per axis
Servosettings for SingleCopter */
#define SINGLECOPTER_YAW {1, 1, 1, 1} // Left, Right,Front, Rear
#define SINGLECOPTER_SERVO {1,-1, 1,-1} // Pitch, Pitch, Roll, Roll

/* Servosettings for DualCopter */
#define DUALCOPTER_SERVO {1,1} //Pitch, Roll
/* Use SERVO_OFFSET and SERVO_RATES in Heli and Airplane section for centering and endpoints */
/***************
***************
SECTION 3 - RC SYSTEM SETUP
***************
***************

/* note: no need to uncomment something in this section if you use a standard receiver */

******************************************************************************
****** special receiver types
******************************************************************************

/ppm_sum_receiver
****************************************************************************/

/* 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 PITCH,YAW,THROTTLE,ROLL,AUX1,AUX2,AUX3,AUX4,8,9,10,11 SERIAL_SUM_PPM //For Graupner/Spektrum
   //define ROLL,PITCH,THROTTLE,YAW,AUX1,AUX2,AUX3,AUX4,8,9,10,11 SERIAL_SUM_PPM //For Robe/Hitec/Futaba
   //define ROLL,PITCH,YAW,THROTTLE,AUX1,AUX2,AUX3,AUX4,8,9,10,11 SERIAL_SUM_PPM //For Multiplex
   //define PITCH,ROLL,THROTTLE,YAW,AUX1,AUX2,AUX3,AUX4,8,9,10,11 SERIAL_SUM_PPM //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

/ppk_satsatellite_reciever
******************************************************************************/

/* 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 SPEK_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 overriden 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.
   //   Un-Comment for Spektrum Satellite Bind Support. Code is ~420 bytes smaller without it.
   //define SPEK_BIND
   //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 at RX1 only (Serial 1). You have to invert the S-Bus-Serial Signal e.g. with a Hex-Inverter like IC SN74 LS 04 */
   //#define SBUS
   //******************************************************************************
   // RC signal from the serial port via Multiwii Serial Protocol **********/
   //define RCSERIAL
   //******************************************************************************
   // SECTION 4 - ALTERNATE CPUs & BOARDS
   //*********/
   //*********/
   //*********/
/// 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
  */
  //define A32U4ALLPINS

/**
  * PWM Setup
  */
  //define HWPWM6

/**
  * Aux 2 Pin
  */
  //define RCAUX2PINRXO
  //define RCAUX2PIND17

/**
  * Buzzer Pin
  */
  //define DBBUZZER

/**
  * Promicro version related
  */
  //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 V_BATPIN A0 // instead of A3 // Analog PIN 3

/#define LEDPIN_PINMODE pinMode (A1, OUTPUT); // use A1 instead of d13
/#define LEDPIN_TOGGLE PINC |= 1<<1; // PINB |= 1<<5;
// switch LEDPIN state (digital PIN 13)
/#define LEDPIN_OFF PORTC &= ~(1<<1); // PORTB &= ~(1<<5);
/#define LEDPIN_ON PORTC |= 1<<1; // was PORTB |= (1<<5);

/#define BUZZERPIN_PINMODE pinMode (A2, OUTPUT); // use A2 instead of d8
/#define BUZZERPIN_ON PORTC |= 1<<2 // PORTB |= 1;
/#define BUZZERPIN_OFF PORTC &= ~(1<<2); // PORTB &= ~1;

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

/* interleaving delay in micro seconds between 2 readings WMP/NK in a WMP+NK config
if the ACC calibration time is very long (20 or 30s), try to increase this delay up to 4000
it is relevent only for a conf with NK */
#define INTERLEAVING_DELAY 3000
/* 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

 Gyro filters

*******************    Lowpass filter for some gyros
********************************/
// #define MPU6050_LPF_20HZ
// #define MPU6050_LPF_10HZ
// #define MPU6050_LPF_5HZ     // Use this only in extreme cases, rather change motors and/or props

******                  Gyro smoothing
*******************************************************************************/

/* GYRO_SMOOTHING. In case you cannot reduce vibrations _and_ _after_ you have tried the low pass filter options, you may try this gyro smoothing via averaging. Not suitable for multicopters!
   Good results for helicopter, airplanes and flying wings (foamies) with lots of vibrations. */
  // #define GYRO_SMOOTHING {20, 20, 3}    // (*) separate averaging ranges for roll, pitch, yaw

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

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

  // #define MMGYRO                   // Active Moving Average Function for Gyros
  // #define MMGYROVECTORLENGHT 10    // Lenght of Moving Average Vector

  /* Moving Average ServoGimbal Signal Output */
  // #define MMSERVOGIMBAL           // Active Output Moving Average Function for Servos Gimbal
  // #define MMSERVOGIMBALVECTORLENGHT 32   // Lenght of Moving Average Vector

/******************    AP FlightMode
*******************************************************************************/

/* Pseudo-derivative controller for level mode (experimental)
   Additional information:
  // #define LEVEL_PDF

*******************************************************************************/
/* Gyrocalibration will be repeated if copter is moving during calibration. */
#define GYROCALIBRATIONFAILSAFE

/************************
AP FlightMode
************************/

/* Temporarily Disables GPS_HOLD_MODE to be able to adjust the Hold-position when moving the sticks. */
#define AP_MODE 20 // 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 // 

/******
Failsafe settings
*******/

/* Failsafe check pulses on four main control channels CH1-CH4. If the pulse is missing or below 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 or nunchuk 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 guard time the RC control is returned to normal. */
#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

/********************
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
#define LANDING_LIGHTS_AUTO_ALTITUDE 50

/* 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

/** Disable WMP power pin *****************************/
/* disable use of the POWER PIN */
#define DISABLE_POWER_PIN

/** TX-related *****************************/
/* introduce a deadband around the stick center
   Must be greater than zero, comment if you don't want a deadband on roll, pitch and yaw */
#define DEADBAND 10

/* defines the neutral zone of throttle stick during altitude hold, default setting is 
   +/-20 uncomment and change the value below if you want to change it. */
//#define ALT_HOLD_THROTTLE_NEUTRAL_ZONE 20

/* GPS using a SERIAL port 
   only available on MEGA boards (this might be possible on 328 based boards in the future) 
   if enabled, define here the Arduino Serial port number and the UART speed 
   note: only the RX PIN is used, the GPS is not configured by multiwii 
   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 
//#define GPS_BAUD 57600 
#define GPS_BAUD 115200

/* 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 
   With UBLOX you don't have to use GPS_FILTERING in multiwii code !!! */

//#define NMEA
//#define UBLOX

//#define INIT_MTK_GPS       // initialize MTK GPS for using selected speed, 5Hz update rate and GGA & RMC sentence

//#define GPS_PROMINI_SERIAL 57600 // Will Autosense if GPS is connected when ardu boots

/* 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 */

//#define I2C_GPS


/* get GPS data from Tiny-GPS */

//#define TINY_GPS

/* get sonar data from Tiny-GPS */

//#define TINY_GPS_SONAR

/* GPS data readed from Misio-OSD - GPS module connected to OSD, and Multiwii read GPS data from OSD - tested and working OK ! */

//#define GPS_FROM_OSD

/* 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

//#define USE_MSP_WP                        //Enables the MSP_WP command, which is used by WinGUI to display and log Home and Poshold positions

//#define DONT_RESET_HOME_AT_ARM             // HOME position is reset at every arm, uncomment it to prohibit it (you can set home position with GyroCalibration)

/* GPS navigation can control the heading */

#define NAV_CONTROLS_HEADING true      // copter faces toward the navigation point, maghold must be enabled for it
#define NAV_TAIL_FIRST false       // true - copter comes in with tail first
#define NAV_SET_TAKEOFF_HEADING true     // true - when copter arrives to home position it rotates it's head to takeoff direction
/* 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) */
   // #define MAG_DECLINATION 3.96f  //For Budapest Hungary.  
   #define MAG_DECLINATION 0.0f

#define GPS_FILTERING  // add a 5 element moving average filter to GPS coordinates, helps eliminate gps noise but adds latency comment out to disable
   #define GPS_LOW_SPEED_D_FILTER  // below .5m/s speed ignore D term for POSHOLD_RATE, theoretically this also removed D term induced noise comment out to disable
   #define GPS_WP_RADIUS 200  // if we are within this distance to a waypoint then we consider it reached (distance is in cm)
   #define NAV_SLEW_RATE 30  // Adds a rate control to nav output, will smoothen out nav angle spikes

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

/*****
*/
#uncomment this line if you plan to use a LCD or OLED */
   // #define LCD_CONF

/* to include setting the aux switches for AUX1 -> AUX4 via LCD */ //to review (activate[] is now 16 bit long)
   // #define LCD_CONF_AUX

/* if program gets too large (>32k), need to exclude some functionality */
/* uncomment to suppress some unwanted aux3 aux4 items in config menu (only useful if LCD_CONF_AUX is enabled) */
   // #define SUPPRESS_LCD_CONF_AUX34

/****************************
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 OLED_I2C_128x64 // I2C LCD: OLED
http://www.multiwii.com/forum/viewtopic.php?f=7&t=1350

/******************************   Logo settings
***********************************/
/#define SUPPRESS_OLED_I2C_128x64LOGO  // suppress display of OLED logo to save memory

/* style of display - AUTODETECTED via LCD_ setting - only activate to overwrite defaults */
/#define DISPLAY_2LINES
/#define DISPLAY_MULTILINE
/#define MULTILINE_PRE 2 // multiline configMenu # pref lines
/#define MULTILINE_POST 6 // multiline configMenu # post lines

********************************    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'

/* To use an LCD03 for configuration:
http://www.robot-electronics.co.uk/htm/Lcd03tech.htm
Remove the jumper on its back to set i2c control.
VCC to +5V VCC (pin1 from top)
SDA - Pin A4 Mini Pro - Pin 20 Mega (pin2 from top)
SCL - Pin A5 Mini Pro - Pin 21 Mega (pin3 from top)
GND to Ground (pin4 from top)*/

/* To use an Eagle Tree Power Panel LCD for configuration:
White wire to Ground
Red wire to +5V VCC (or to the WMP power pin, if you prefer to reset everything on the bus when WMP resets)*/
Yellow wire to SDA - Pin A4 Mini Pro - Pin 20 Mega
Brown wire to SCL - Pin A5 Mini Pro - Pin 21 Mega */

/* Cat's whisker LCD TEXTSTAR LCD
Please note this display needs a full 4 wire connection to (+5V, Gnd, RXD, TXD )
Configure display as follows: 115K baud, and TTL levels for RXD and TXD, terminal mode
NO rx / tx line reconfiguration, use natural pins.
The four buttons sending 'A', 'B', 'C', 'D' are supported for configuration navigation and request of telemetry pages 1-4 */

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

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

/* to monitor system values (battery level, loop time etc. with LCD enable this note: for now you must send single characters to request different pages
Buttons toggle request for page on/off
The active page on the LCD does get updated automatically
Easy to use with Terminal application or display like LCD - if available uses the 4 preconfigured buttons to send 'A', 'B', 'C', 'D' */
/****
/****
/****
/****

/#define LCD_TELEMETRY

/* to enable automatic hopping between a choice of telemetry pages uncomment this.
This may be useful if your LCD has no buttons or the sending is broken
hopping is activated and deactivated in unarmed mode with throttle=low & roll=left & pitch=forward
set it to the sequence of telemetry pages you want to see
2 line displays support pages 1-9
multiline displays support pages 1-5 */
/****
/****
/****
/****

/#define LCD_TELEMETRY_AUTO "123452679" // pages 1 to 7 in ascending order
/****
/****
/****
/****

/#define LCD_TELEMETRY_AUTO "212232425262729" // strong emphasis on page 2

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

/* same as above, but manual stepping sequence; requires stick input (throttle=low & roll=right & pitch=forward) to
step through each defined telemetry page
First page of the sequence gets loaded at startup to allow non-interactive display */
    //#define LCD_TELEMETRY_STEP "0123456789" // should contain a 0 to allow switching off. First page of sequence gets loaded upon startup

/* if program gets too large (>32k), need to exclude some functionality uncomment to suppress some unwanted telemetry pages (only useful if telemetry is enabled) */
    //#define SUPPRESS_TELEMETRY_PAGE_1
    //#define SUPPRESS_TELEMETRY_PAGE_2
    //#define SUPPRESS_TELEMETRY_PAGE_3
    //#define SUPPRESS_TELEMETRY_PAGE_4
    //#define SUPPRESS_TELEMETRY_PAGE_5
    //#define SUPPRESS_TELEMETRY_PAGE_6
    //#define SUPPRESS_TELEMETRY_PAGE_7
    //#define SUPPRESS_TELEMETRY_PAGE_8
    //#define SUPPRESS_TELEMETRY_PAGE_9

/********************************************************************/
****
Buzzer
****/
/********************************************************************/
    //#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-Arcraft 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
#define VBATLEVEL1_3S 107 // (*) 10,7V
#define VBATLEVEL2_3S 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
/* enable monitoring of the power consumption from battery (think of mAh)
   allows to set alarm value in GUI or via LCD
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
/* PLEVELSCALE is the step size you can use to set alarm */
#define PLEVELSCALE 50 // if you change this value for other granularity, you must search for comments in code to change accordingly
/* larger PLEVELDIV will get you smaller value for power (mAh equivalent) */
#define PLEVELDIV 5000 // (*) default for soft - if you lower PLEVELDIV, beware of overrun in uint32 pMeter
#define PLEVELDIVSOFT PLEVELDIV // for soft always equal to PLEVELDIV; for hard set to 5000
#define PSENSORNULL 510 // (*) set to analogRead() value for zero current; for I=0A my sensor gives 1/2 Vss; that is approx 2.49Volt;
#define PINT2mA 13 // (*) for telemetry display: one integer step on arduino analog translates to mA (example 4.9 / 37 * 100

/* uncomment to disable the altitude hold feature.
* This is useful if all of the following apply
* + you have a baro
* + want altitude readout
* + do not use altitude hold feature
* + want to save memory space */
//#define SUPPRESS_BARO_ALTHOLD

/* 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

************ SECTION 7 - TUNING & DEVELOPER ************
************

EXperimental: force a stable, fixated (high) cycle time

when activated, the displayed cycle time in GUI will not be correct.
Tunable via LCD config menu.
value of 0 turns the feature off.

#define CYCLETIME_FIXATED 9000 // (*)

************ special ESC with extended range [0-2000] microseconds ************

#define EXT_MOTOR_RANGE

************ 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 */
#ifndef 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 (assigned to PPM or SPECTRUM CH9 on copter configuration)
SERVO5 = pin 12 (assigned to PPM or SPECTRUM CH10 on copter configuration)
this option disable other software PWM's for servos - only five hardware controlled servos available */
define MEGA_HW_PWM_SERVOS

 IMU complimentary filter tuning */

/* Set the Low Pass Filter factor for ACC */
Increasing this value would reduce ACC noise (visible in GUI), but would increase ACC lag time
Comment this out if you want to set a specific coeff (non default)*/
define ACC_LPF_FACTOR 100

/* Set the Low Pass Filter factor for Magnetometer */
Increasing this value would reduce Magnetometer noise (not visible in GUI), but would increase Magnetometer lag time
Comment this out if you want to set a specific coeff (non default)/
/
/* Set the Gyro Weight for Gyro/Acc complementary filter
   Increasing this value would reduce and delay Acc influence on the
output of the filter
Comment this out if you want to set a specific coeff (non default)/
/
#define MG_LPF_FACTOR 4

/* Set the Gyro Weight for Gyro/Magnetometer complementary filter
 Increasing this value would reduce and delay Magnetometer influence
on the output of the filter
Comment this out if you want to set a specific coeff (non default)/
/
#define GYR_CMPF_FACTOR 400.0f

#define GYR_CMPFM_FACTOR 200.0f

/*****************************/
/**** diagnostics ************/
/*****************************/

/* to log values like max loop time and others to come
  logging values are visible via LCD config
  set to 2, if you want more values,
  set to 3 for 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

/* to add debugging code
 not needed and not recommended for normal operation
 will add extra code that may slow down the main loop or make copter
non-flyable */
/
#define DEBUG

/* Use this to trigger LCD configuration without a TX - only for
debugging - do NOT fly with this activated */
/
#define LCD_CONF_DEBUG

/* Use this to trigger telemetry without a TX - only for debugging - do
NOT fly with this activated */
/
#define LCD_TELEMETRY_DEBUG    //This form stays on the screen
#define LCD_TELEMETRY_AUTO 6  //This form rolls between all screens,
LCD_TELEMETRY_AUTO must also be defined.
/
#define LCD_TELEMETRY_DEBUG 6  //This form stays on the screen
specified.

/* Enable string transmissions from copter to GUI */
/
#define DEBUGMSG

/*****************************/
/**** ESCs calibration ************/
/* 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 ****
/* frequenies 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 667 // to step to next telemetry page 667 <=> 2s
#define PSENSORFREQ 6
#define VBATFREQ PSENSORFREQ        // to read battery voltage - keep equal to PSENSORFREQ unless you know what you are doing

**** 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.
*/
#ifndef COPTERTEST
#define COPTERTEST 1
#endif