// 250209 PWMoutAGLDABFoldWing

//copyright Kazuhiko Kakuta 20250125

#include <Servo.h>

#include <EEPROM.h>

#include "src/PPMReader/PPMReader.h"// <PPMReader.h>

//#include <InterruptHandler.h>// 2022/01/27 Delete for more good move

// PPMtoPWM

int interruptPin = 2;

int channelAmount = 8;

PPMReader ppm(interruptPin, channelAmount);

int servo\_1\_pin = 3;

int servo\_2\_pin = 4;

int servo\_3\_pin = 5;

int servo\_4\_pin = 6;

int servo\_5\_pin = 7;

int servo\_6\_pin = 8;

int servo\_7\_pin = 9;

int servo\_8\_pin = 10;

volatile int ch3value = 950;// Ch3 ESC

volatile int ch1value = 1500;// Ch1 Aileron

volatile int ch2value = 1500;// Ch2 Elevator

volatile int ch4value = 1500;//Ch4

volatile int ch5value = 1500;// Ch5 Wing Folding On-Off-Stoop switch D

volatile int ch6value = 1500;// Ch6 Wing Folding Time trim 0-60 time WFTT Aux5 RtVolume

volatile int ch7value = 1500;//Ch7 Wing Folding trim WFtrim Aux4 LtVolume

volatile int ch8value = 1500;//Ch8

volatile int RtServo =2000;

volatile int LtServo =1000;

volatile int RtVtailS =1500;

volatile int LtVtailS =1500;

volatile int WFTime =300 ;// Wing Folding Time WFT  1/4phase (msec)125-618mSec

volatile int WFTT =30;// Wing Folding Time trim WFTT 0-60 time

volatile int WFTi = 9000; //300\*30

volatile int WFtrim =0;//Wing fording degree 0-1000

Servo servo\_1, servo\_2, servo\_3, servo\_4, servo\_5, servo\_6, servo\_7, servo\_8; // create servo object to control a servo

//GLDAB by Arduino

volatile int PreGMS = 1300;// PreGlideMotorSpeed =900 + pgms\*3

volatile int pgms = 0;// pgms = (PreGlideMotorSpeed -900) / 3

             // int a = 0; EEPROM.write(a, pgms); pgms = EEPROM.read(a);

volatile int motorstop = 900;

int val = 1;

int val2 = 1;

int a = 0;

volatile int flag = 0; // do wright of pgms in EEPROM

volatile int flag2 = 0; // do PreGM

volatile int flag3 = 0; // after setting PreGM do not only first PreGM

const int LED1 =13;// LED on Arduino board

const int Hole =11; // Hole Senser Detect pin set on 11pin D11

const int WFold =12; // Hole Senser for Wing Folding Detect pin set on 12pin D12

void setup() {

Serial.begin(9600);

pinMode(servo\_1\_pin, OUTPUT);

pinMode(servo\_3\_pin, OUTPUT);

pinMode(servo\_4\_pin, OUTPUT);

pinMode(servo\_5\_pin, OUTPUT);

pinMode(servo\_6\_pin, OUTPUT);

pinMode(servo\_7\_pin, OUTPUT);

pinMode(servo\_8\_pin, OUTPUT);

servo\_1.attach(servo\_1\_pin);//output pin No

servo\_2.attach(servo\_2\_pin);//output pin No

servo\_3.attach(servo\_3\_pin);//output pin No

servo\_4.attach(servo\_4\_pin);//output pin No

servo\_5.attach(servo\_5\_pin);//output pin No

servo\_6.attach(servo\_6\_pin);//output pin No

servo\_7.attach(servo\_7\_pin);//output pin No

servo\_8.attach(servo\_8\_pin);//output pin No

pinMode(LED1, OUTPUT);// Blink LED

pinMode(Hole, INPUT);//pin11 Hole senser detect pin set input

pinMode(WFold, INPUT);//pin12 Hole senser detect pin set input for WingFolding

pgms = EEPROM.read(a);

PreGMS =900 + pgms\*3;

delay(2000);//Avoid abnormal positions at startup-wait 2 second until RX starts231004

}

void loop() {

//Serial.print("flag-");

//Serial.println(flag);

ch3value = ppm.rawChannelValue(3);//Ch3

if (( ch3value > 1950) && (flag == 0)){

digitalWrite(LED1, HIGH);  //RedLED ON

servo\_3.writeMicroseconds(motorstop); //Motor stop pre ESC setting

//Need ESC calibration before connect new GLDAB

delay(3000); //if LED on, then set throttle stick down max low within 2second

digitalWrite(LED1, LOW); //  LED Off

ch3value = ppm.rawChannelValue(3);//Ch3

if ( ch3value < 1040){

delay(2000);//Wait 2sec

digitalWrite(LED1, HIGH);//  while LED ON set stick at PreGlideMotorSpeed

for (int s = 0 ; s < 1500 ; s++ ){

ch3value = ppm.rawChannelValue(3);//Ch3

servo\_3.writeMicroseconds( ch3value );

PreGMS = ch3value; //after 4sec set motor speed now to PreGlideMotorSpeed

pgms = (PreGMS - 900) / 3;

EEPROM.write(a, pgms);// memory pgmr in EEPROM

//Serial.print("setPreGMS-");

//Serial.println(PreGMS);

//Serial.print("s-");

//Serial.println(s);

}

digitalWrite(LED1, LOW);    // turn the LED off and motor stop

delay(1000);

for(int k = 0 ; k < 4  ; k++){

servo\_3.writeMicroseconds(motorstop);//Stop motor

//Serial.print("k-");

//Serial.println(k);

digitalWrite(LED1, HIGH);//End of setting of PreGlideMotorSpeed

delay(500);

digitalWrite(LED1, LOW);

delay(500);

}

flag = 1 ;

}

}else

{

// Flapping loop

flag = 1;

ch5value = ppm.rawChannelValue(5);//Ch5 Wing Folding on-Off -Stoop switch

//Flapping with Fold wing at up stroke

if (ch5value >1700) {// Flap with folding wing on flap cycle ON

ch1value = ppm.rawChannelValue(1);//Ch1 Aileron

ch2value = ppm.rawChannelValue(2);//Ch2 Elevator

ch3value = ppm.rawChannelValue(3);//Ch3 ESC

ch4value = ppm.rawChannelValue(4);//Ch4 inverted V Tail

ch5value = ppm.rawChannelValue(5);//Ch5 Wing Folding- no Fold- Stoop Switch E

ch6value = ppm.rawChannelValue(6);//Ch6 Wing Folding Time trim WFTT Aux5

ch7value = ppm.rawChannelValue(7);//Ch7 Wing Folding trim WFtrim Aux4

ch8value = ppm.rawChannelValue(8);//Ch8

WFTime=-0.821\*ch3value+1768;

WFTT =0.01\*ch6value-10; // Wing Folding Time trim WFTT

WFTi= WFTime\*WFTT;

WFtrim = ch7value - 1000;

RtServo = 2\* ch1value -2000;

LtServo = 2\* ch1value -1000;

RtServo = constrain (RtServo, 900,2100);

LtServo = constrain (LtServo, 900,2100);

RtVtailS = ch4value + (-ch2value + 1500);

LtVtailS = ch4value + (ch2value - 1500);

RtVtailS = constrain (RtVtailS, 900,2100);

LtVtailS = constrain (LtVtailS, 900,2100);

//Serial.print("ch7value-");

//Serial.print(ch7value);

//Serial.print("RtServo-");

//Serial.print(RtServo);

//Serial.print("LtServo-");

//Serial.println(LtServo);

// motor move by now throttle

 servo\_1.writeMicroseconds( RtServo);//Folding WingRt

 servo\_2.writeMicroseconds( RtVtailS );//RtTailServo

 servo\_3.writeMicroseconds( ch3value );//ESC

 servo\_4.writeMicroseconds( LtServo ); //Folding WingLt

 servo\_5.writeMicroseconds( LtVtailS );//LtTailServo

 servo\_6.writeMicroseconds( ch6value );

 servo\_7.writeMicroseconds( ch7value );

 servo\_8.writeMicroseconds( ch8value );

val2 = digitalRead(WFold); //pin12 Hole senser detect pin set input for WingFolding

if (( val2 == 0) && (ch3value > 1080)){//If the Ch3 value is less than 1080msec, the wing folding will not occur

for(int i = 0 ; i < 18000 ; i++) {

ch1value = ppm.rawChannelValue(1);//Ch1 Aileron

ch2value = ppm.rawChannelValue(2);//Ch2 Elevator

ch3value = ppm.rawChannelValue(3);//Ch3 ESC

ch4value = ppm.rawChannelValue(4);//Ch4

ch5value = ppm.rawChannelValue(5);//Ch5 Wing Folding On-Off -stoop Switch E

ch6value = ppm.rawChannelValue(6);//Ch6 Wing Folding Time trim WFTT left Lever

ch7value = ppm.rawChannelValue(7);//Ch7 Wing Folding trim WFtrim right Lever

ch8value = ppm.rawChannelValue(8);//Ch8

WFTime=-0.821\*ch3value+1768;

WFTT =0.01\*ch6value-10; // Wing Folding Time trim WFTT

WFTi= WFTime\*WFTT;

if (i > WFTi){

  i = 18000;}

WFtrim= ch7value-1000;

RtServo =2\* ch1value -2000 + WFtrim;

LtServo =  2\* ch1value -1000 - WFtrim;

RtServo = constrain (RtServo, 900,2100);

LtServo = constrain (LtServo, 900,2100);

RtVtailS = ch4value + (-ch2value + 1500);

LtVtailS = ch4value + (ch2value - 1500);

RtVtailS = constrain (RtVtailS, 900,2100);

LtVtailS = constrain (LtVtailS, 900,2100);

// motor move by now throttle

 servo\_1.writeMicroseconds( RtServo);

 servo\_2.writeMicroseconds( RtVtailS );

 servo\_3.writeMicroseconds( ch3value );

 servo\_4.writeMicroseconds( LtServo );

 servo\_5.writeMicroseconds( LtVtailS );

 servo\_6.writeMicroseconds( ch6value );

 servo\_7.writeMicroseconds( ch7value );

 servo\_8.writeMicroseconds( ch8value );

}

}

}

//Flapping without Fold wing at up stroke

if ((ch5value >1300) && ( ch5value <1700)){

ch1value = ppm.rawChannelValue(1);//Ch1 Aileron

ch2value = ppm.rawChannelValue(2);//Ch2 Elevator

ch3value = ppm.rawChannelValue(3);//Ch3 ESC

ch4value = ppm.rawChannelValue(4);//Ch4

ch5value = ppm.rawChannelValue(5);//Ch5 Wing Folding On-Off

ch6value = ppm.rawChannelValue(6);//Ch6 WingFolding duration

ch7value = ppm.rawChannelValue(7);//Ch7

ch8value = ppm.rawChannelValue(8);//Ch8

RtServo = 2\* ch1value -2000;

LtServo = 2\* ch1value -1000;

RtServo = constrain (RtServo, 900, 2100);

LtServo = constrain (LtServo, 900, 2100);

RtVtailS = ch4value + (- ch2value +1500);

LtVtailS = ch4value + (ch2value - 1500);

RtVtailS = constrain (RtVtailS, 900,2100);

LtVtailS = constrain (LtVtailS, 900,2100);

// motor move by now throttle

 servo\_1.writeMicroseconds( RtServo );

 servo\_2.writeMicroseconds( RtVtailS );

 servo\_3.writeMicroseconds( ch3value );

 servo\_4.writeMicroseconds( LtServo);

 servo\_5.writeMicroseconds( LtVtailS );

 servo\_6.writeMicroseconds( ch6value );

 servo\_7.writeMicroseconds( ch7value );

 servo\_8.writeMicroseconds( ch8value );

}

//Flapping in Stoop with Wing Folding

if (ch5value <1300){// Stoop

ch1value = ppm.rawChannelValue(1);//Ch1 Aileron

ch2value = ppm.rawChannelValue(2);//Ch2 Elevator

ch3value = ppm.rawChannelValue(3);//Ch3 ESC

ch4value = ppm.rawChannelValue(4);//Ch4

ch5value = ppm.rawChannelValue(5);//Ch5 Wing Folding On-Off Switch E

ch6value = ppm.rawChannelValue(6);//Ch6 Wing Folding Time trim WFTT left Lever

ch7value = ppm.rawChannelValue(7);//Ch7 Wing Folding trim WFtrim right Lever

ch8value = ppm.rawChannelValue(8);//Ch8

WFTime=-0.821\*ch3value+1768;

WFTT =0.01\*ch6value-10; // Wing Folding Time trim WFTT

WFTi= WFTime\*WFTT;

WFtrim= ch7value-1000;

RtServo =2\* ch1value -2000 + WFtrim;

LtServo =  2\* ch1value -1000 - WFtrim;

RtServo = constrain (RtServo, 900,2100);

LtServo = constrain (LtServo, 900,2100);

RtVtailS = ch4value + (-ch2value + 1500);

LtVtailS = ch4value + (ch2value - 1500);

RtVtailS = constrain (RtVtailS, 900,2100);

LtVtailS = constrain (LtVtailS, 900,2100);

// motor move by now throttle

 servo\_1.writeMicroseconds( RtServo);

 servo\_2.writeMicroseconds( RtVtailS );

 servo\_3.writeMicroseconds( ch3value );

 servo\_4.writeMicroseconds( LtServo );

 servo\_5.writeMicroseconds( LtVtailS );

 servo\_6.writeMicroseconds( ch6value );

 servo\_7.writeMicroseconds( ch7value );

 servo\_8.writeMicroseconds( ch8value );

}

// Glide lock with stretched Wing

if (ch5value >1300){ //GLDAB act on Wing open

//Serial.print("-");

//Serial.println(ch3value);//Set Throttle Min <920 Max >1980 by serial monitor

if ( flag3 == 0){goto label;}

if (( ch3value < 950 ) && (flag2 == 0 )){

digitalWrite(LED1, HIGH);

pgms = EEPROM.read(a);

PreGMS =900 + pgms\*3;

for(int i = 0 ; i < 6000 ; i++) {  //10000=about 0.72or 0.66second

//ch3value = ppm.rawChannelValue(3);//Ch3

ch1value = ppm.rawChannelValue(1);//Ch1

ch2value = ppm.rawChannelValue(2);//Ch2

ch4value = ppm.rawChannelValue(4);//Ch4

ch5value = ppm.rawChannelValue(5);//Ch5

ch6value = ppm.rawChannelValue(6);//Ch6

ch7value = ppm.rawChannelValue(7);//Ch7

ch8value = ppm.rawChannelValue(8);//Ch8

RtServo = 2\* ch1value -2000 ;

LtServo = 2\* ch1value -1000 ;

RtServo = constrain (RtServo, 900,2100);

LtServo = constrain (LtServo, 900, 2100);

RtVtailS = ch4value + (- ch2value + 1500);

LtVtailS = ch4value + (ch2value - 1500);

RtVtailS = constrain (RtVtailS, 900,2100);

LtVtailS = constrain (LtVtailS, 900,2100);

 servo\_3.writeMicroseconds(PreGMS);

 servo\_1.writeMicroseconds( RtServo );

 servo\_2.writeMicroseconds( RtVtailS );

 servo\_4.writeMicroseconds( LtServo );

 servo\_5.writeMicroseconds( LtVtailS );

 servo\_6.writeMicroseconds( ch6value );

 servo\_7.writeMicroseconds( ch7value );

 servo\_8.writeMicroseconds( ch8value );

//Serial.print("PreGMS--");

//Serial.println(PreGMS);

//Serial.print("i-");

//Serial.println(i);

val = digitalRead(Hole); //read pin6D6 magnet holesenser

//Serial.print("val-");// Not delete “Serial.print("val-")” and “Serial.println(val);”

//Serial.println(val);//

if ( val == 0){ i = 6000;}// When a magnet is detected, Val=0 and the motor stops

   // Same number line 146 "i < 6000"

ch3value = ppm.rawChannelValue(3);//Ch3

if ( ch3value > 951 ) { i = 6000;}  // Same number line 146 "i < 6000"

}

digitalWrite(LED1, LOW);

label:

flag2=1;

flag3=1;

}

ch3value = ppm.rawChannelValue(3);//Ch3

if ( ch3value > 951 ){flag2 = 0;}

}

// Glide lock with Stooped Wing

if (ch5value <1300){// Stoop GLDAB act on Folding Wing

//Serial.print("-");

//Serial.println(ch3value);//Set Throttle Min <920 Max >1980 by serial monitor

if ( flag3 == 0){goto labe2;}

if (( ch3value < 950 ) && (flag2 == 0 )){

digitalWrite(LED1, HIGH);

pgms = EEPROM.read(a);

PreGMS =900 + pgms\*3;

for(int i = 0 ; i < 6000 ; i++) {  //10000=about 0.66or0.72second

//ch3value = ppm.rawChannelValue(3);//Ch3

ch1value = ppm.rawChannelValue(1);//Ch1

ch2value = ppm.rawChannelValue(2);//Ch2

ch4value = ppm.rawChannelValue(4);//Ch4

ch5value = ppm.rawChannelValue(5);//Ch5

ch6value = ppm.rawChannelValue(6);//Ch6

ch7value = ppm.rawChannelValue(7);//Ch7

ch8value = ppm.rawChannelValue(8);//Ch8

WFTime=-0.821\*ch3value+1768;

WFTT =0.01\*ch6value-10; // Wing Folding Time trim WFTT

WFTi= WFTime\*WFTT;

WFtrim= ch7value-1000;

RtServo = 2\* ch1value -2000 + WFtrim;

LtServo =  2\* ch1value -1000 - WFtrim;

RtServo = constrain (RtServo, 900,2100);

LtServo = constrain (LtServo, 900,2100);

RtVtailS = ch4value +(- ch2value + 1500);

LtVtailS = ch4value + (ch2value - 1500);

RtVtailS = constrain (RtVtailS, 900,2100);

LtVtailS = constrain (LtVtailS, 900,2100);

 servo\_3.writeMicroseconds(PreGMS);

 servo\_1.writeMicroseconds( RtServo );

 servo\_2.writeMicroseconds( RtVtailS );

 servo\_4.writeMicroseconds( LtServo );

 servo\_5.writeMicroseconds( LtVtailS );

 servo\_6.writeMicroseconds( ch6value );

 servo\_7.writeMicroseconds( ch7value );

 servo\_8.writeMicroseconds( ch8value );

//Serial.print("PreGMS--");

//Serial.println(PreGMS);

//Serial.print("i-");

//Serial.println(i);

val = digitalRead(Hole); //read pin6D6 magnet holesenser

//Serial.print("val-");// Not delete “Serial.print("val-")” and “Serial.println(val);”

//Serial.println(val);//

if ( val == 0){ i = 6000;}// When a magnet is detected, Val=0 and the motor stops

   // Same number line 146 "i < 6000"

ch3value = ppm.rawChannelValue(3);//Ch3

if ( ch3value > 951 ) { i = 6000;}  // Same number line 146 "i < 6000"

}

digitalWrite(LED1, LOW);

labe2:

flag2=1;

flag3=1;

}

ch3value = ppm.rawChannelValue(3);//Ch3

if ( ch3value > 951 ){flag2 = 0;}

}

}

}//loop