Středoškolská technika 2016

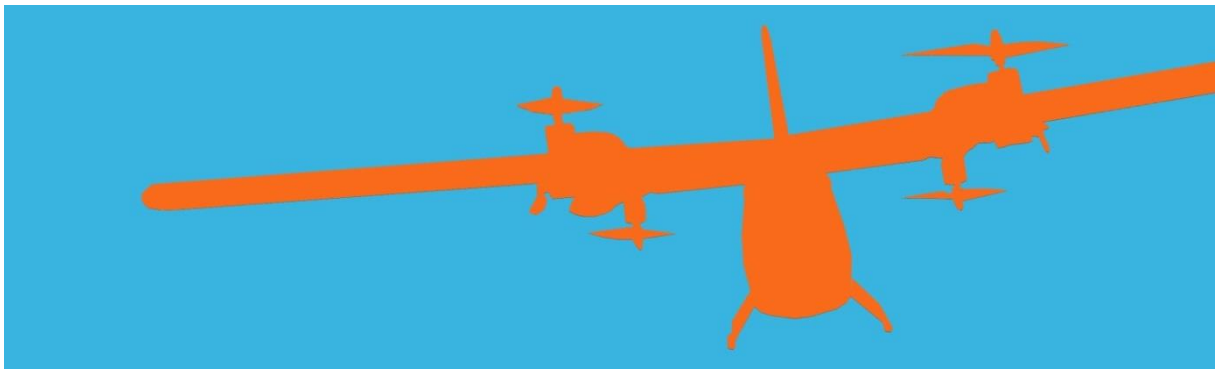
Setkání a prezentace prací středoškolských studentů na ČVUT

QUADROLETADLO

Michal Smutný, Jakub Šašek

Střední průmyslová škola strojnická a Střední odborná škola profesora Švejcara, Plzeň
Klatovská 109, 301 00 Plzeň

Středoškolská odborná činnost



Quadroletadlo

Autor:
MICHAL SMUTNÝ
JAKUB ŠAŠEK
KONZULTANT: ING. JULIUS REGULA OPONENT: ING. PETR HLÁVKA
STŘEDNÍ PRŮMYSLOVÁ ŠKOLA STROJNICKÁ A STŘEDNÍ ODBORNÁ ŠKOLA PROFESORA ŠVEJCARA

evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
PLZEŇSKÝ KRAJ
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Prohlášení

Prohlašujeme, že jsme svou práci vypracovali samostatně, použili jsme pouze podklady (literaturu, SW atd.) uvedené v přiloženém seznamu a postup při zpracování a dalším nakládání s prací je v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) v platném znění.

Poděkování

Tímto bych chtěl poděkovat Ing. Juliu Regulovi za pomoc a podporu při vývoji, Ing. Petru Hlávkovi za zprostředkování finančních prostředků z grantů Evropské unie a také SPŠ strojnické SOŠ profesora Švejcara za možnost vytvořit tento projekt.

Anotace

Našim cílem bylo pokusit se o posunutí možností coptér při jejich komerčním i nekomerčním využití v integrovaném záchranném systému. Díky menším spotřebě energie je quadroletadlo lehčí a může unést více vybavení. Případně může se stejným vybavením vydržet po delší dobu v operačním prostoru.

Klíčová slova

Quadroletadlo, servo, RC model, Arduino, KK2.0, programování, 3D tisk, 3D modelování, quadrocoptera, letadlo, BLDC motory .

Annotation

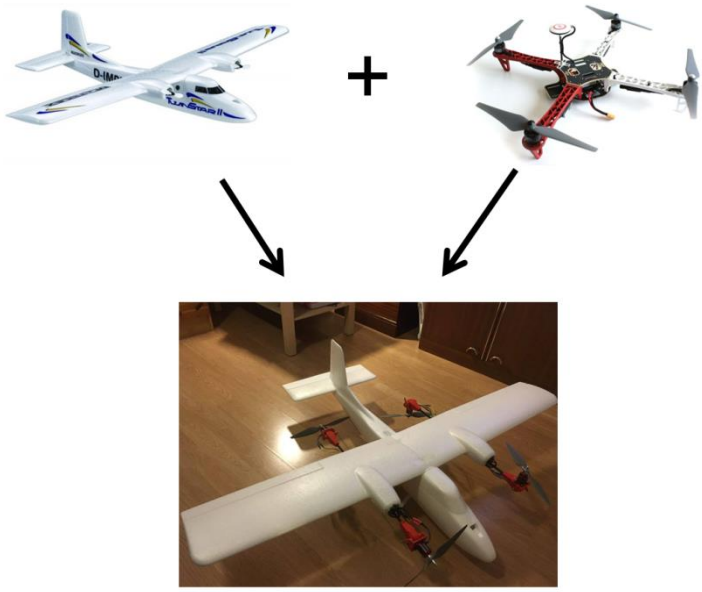
Our intention was to try to move posibilities of copters in comercial and noncomercial usage in IZS. Thanks to low power usage is Quadroplane lighter and can carry more equipment or can stay in the air more time with basic equipment.

Obsah

1. Úvod

Cílem projektu je zkombinovat schopnosti quadrocoptery a letadla prostřednictvím programem řízeného překlápění motorů (obr. 1). Získáme tím multifunkční stroj s možností kolmého startu a visu ve vzduchu na místě, což jsou výhody quadrocoptery a možností šetření energie při využití vztlaku křídel letadla v horizontálním letu. Díky tomu je Quadroletadlo lehčí, tak že může nést více vybavení, nebo se stejnou výbavou uletět větší vzdálenost.

Hlavními komponentami našeho modelu jsou řídící jednotky KK2.0 a Arduino, které se starají o řízení celého systému. K dálkovému bezdrátovému ovládání je využíváno ovládání Devo7, které nám umožňuje řízení na poměrně velké vzdálenosti cca 1,5 km. Pro přenos video signálu je použit FPV systém od firmy Boscoma s úměrným dosahem k ovladači.



Obr. 1 – Princip quadroletadla

2. Vybavení

2.1 Letadlo

Obr. 2 – Původní model

Multiplex Twin Star II (obr. 2) je stavebnice dvoumotorového hornoplošníku s akrobatickými schopnostmi o rozpětí 1 420 mm, vyrobený z odolného materiálu Elapor.

Dvoumotorový RC model Twin Star 2 z ELAPORu navazuje na osvědčeného a velmi oblíbeného polystyrenového předchůdce TwinStar. Oproti němu má nové dvoudílné křídlo vyztužené kompozitovým nosníkem s větší nosnou plochou a opatřené novým profilem, což

zajišťuje lepší letové vlastnosti při nízkých rychlostech. „Dvojka“ TwinStaru má elegantnější tvary trupu a celkově pokročilejší konstrukci.

2.2 Elektronická výbava

2.2.1 Regulační a akční členy

Jako pohonné motory jsme použili 250 W motory Power HD 2835/1038KV (obr. 3).

Obr. 3 – Motor

S těmito motory máme velmi dobré zkušenosti z jiných RC modelů. Tyto motory patří do kategorie BLDC motorů, což jsou stejnosměrné motory s elektronickou komutací, kterou zajišťuje elektronická jednotka motoru přepínající jednotlivá vinutí statoru (nahrazuje mechanický komutátor) v závislosti na požadovaných vlastnostech motoru. V našem případě je to výrobcem doporučený 30 A regulátor, který v mikroprocesoru zpracovává informace o požadovaných otáčkách motoru. Následně na jejich základě generuje data pro spínání výkonových FET tranzistorů. Regulátor dále také díky mikroprocesoru kontroluje teplotu motoru, případné špatné zapojení nebo zkrat.

Pro ovládání řídících ploch modelu jsou použita běžná modelářská serva TG – 9 se silou 1,8 kg/cm (obr. 4).

Obr. 4 – Servo

Pro překlápění motorů byla zvolena jejich silnější verze s titanovými převody se silou 3,2 kg/cm.

2.2.2 Přenosové stanice

Pro přenos signálů řízení jsme použili sedmi kanálovou vysílací soupravu Devo7 (obr. 5).

Obr. 5 – Vysílač

Tento vysílač pracuje v pásmu 2,4 GHz s protokolem DSSS. Bohužel tento protokol má velkou šířku pásma, takže se dá snadno zarušit, pokud se v jeho přítomnosti na této frekvenci přenáší velké objemy dat např.: wi-fi komunikace – jak jsme náhodně zjistili na jedné soutěži.

Pro přenos videa neboli FPV (First-Person View, obr. 6) signálu jsme byli kvůli české legislativě nuceni použít 5,8 GHz vysílač od firmy Boscoma, který dokáže sice přenést relativně hodně dat, ale bohužel se dá snadno zarušit překážkou v přímé cestě signálu. Tomuto zarušení se dá částečně předejít použitím zařízení circularpolarized antenna (v překladu kruhových antén), které jsme použili i na našem modelu. Jak vypovídá jejich název, vlny vysílaného signálu neposílají po přímce, ale stáčí je do kruhu. Zajímavým zjištěním pro nás bylo, že ač je toto vysílání vysokofrekvenční (5,8 GHz), přenášená data se posílají analogově. Což jsme zjistili z technické dokumentace FPV systému.

Obr. 6 – Displej pro First-Person View

2.2.3 Napájení

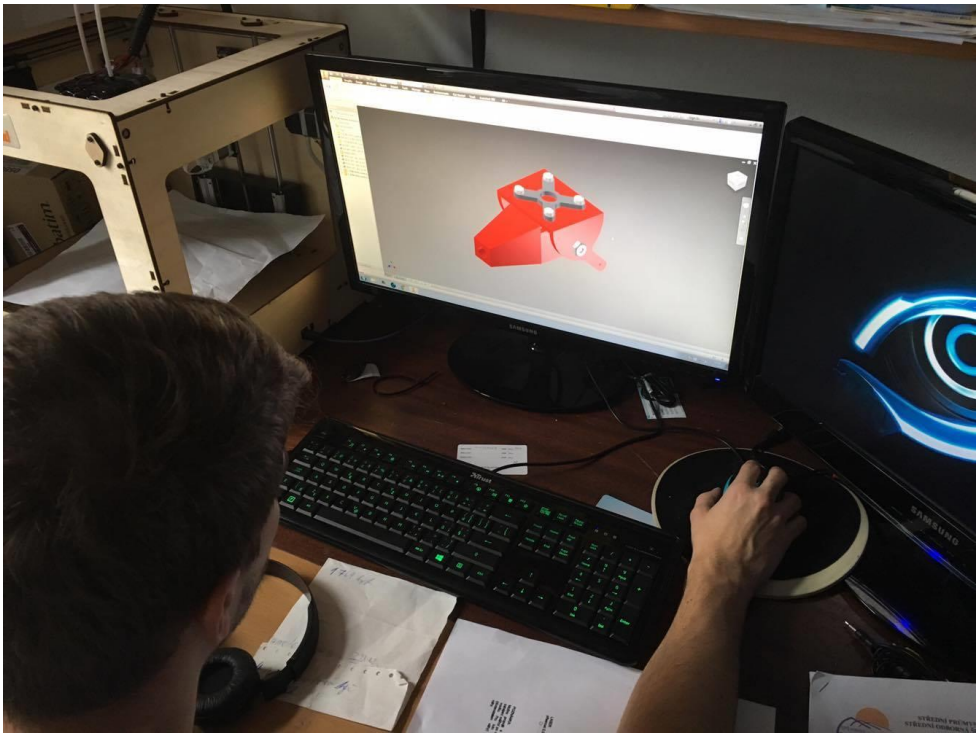
Obr. 7 – Použitá Li-Pol baterie

Pro napájení modelu jsme použili tříčlánkovou Li-Pol (lithium polymerovou) baterii (obr. 7). Baterie tohoto typu jsou velice lehké při zachování velké kapacity a jsou schopny dodávat velké proudy v našem případě se jedná o tříčlánkovou (cca 12 V) baterii s kapacitou 4000 mAh a proudovou zatížitelností 35 C (C = kapacita článku) – to znamená, že naše baterie je schopna stabilně bez poškození dodávat až 140 A (35 C * 4 Ah) a s nabíjecím proudem až 5 C (20 A). Tento nabíjecí proud z vlastní modelářské zkušenosti nedoporučujeme z důvodu ničení baterie a snižování její kapacity, ideální nabíjecí proud je 1 C. Nevýhodnou použitých baterií je však nutnost speciální nabíječky a balancování jednotlivých článků baterie, jejichž napětí nesmí klesnout pod 2,7 V a nesmí stoupnout nad 4,23 V, jinak dojde k jejich nenávratnému zničení. Při přebití nebo překročení maximálních dovolených proudových hodnot (zkrat) může dojít k výbuchu baterie. Časem se také zvětšuje tlak uvnitř baterie a klesá její kapacita. Z mé osobní zkušenosti jsou však námi použité baterie s větší kapacitou od firmy Dualsky, velice odolné vůči těmto problémům.

3. Konstrukce

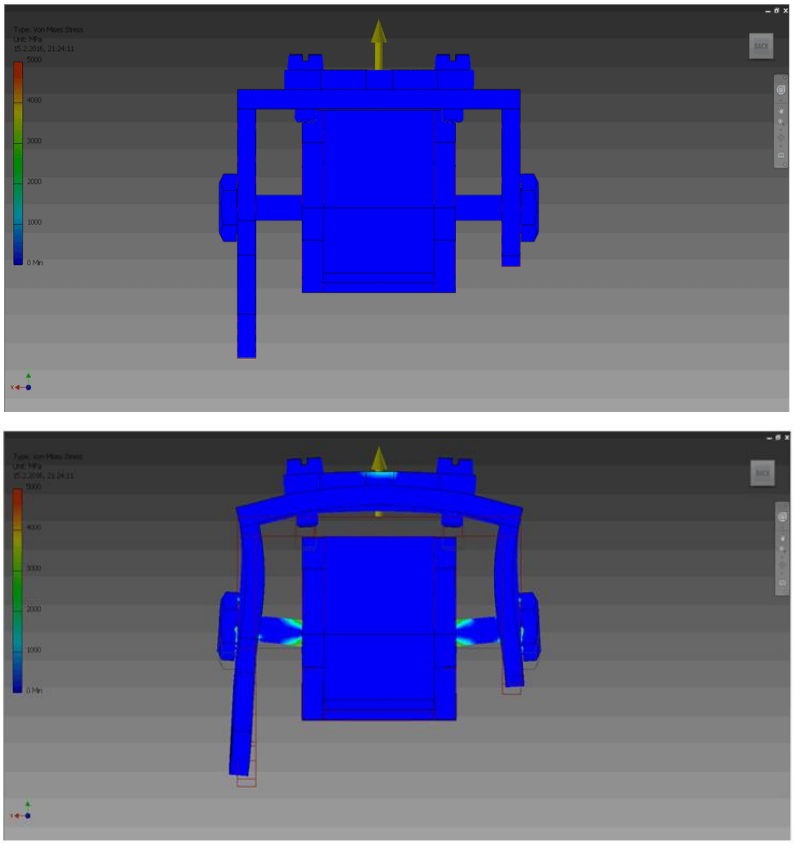
3.1 3D model

Pro vývoj překlápěcího systému motorů (obr. 8) jsme použili program Autodesk Inventor 2015. V něm jsme nejprve postupně vymodelovali části tohoto systému, poté je sestavili do sestavy, na níž jsme si mohli vyzkoušet, zda části modelu splňují naše požadavky. Ověřili jsme, že se o sebe netřou a nedochází k jejich kolizi. Také jsme v programu Inventor ověřovali rozdíly hmotnosti překlápěcího systému motorů při použití různých materiálů (v našem případě hliník versus ABS plast). Pro hliník se hmotnost pohybovala kolem 150 g, zatímco pro ABS plast pouze 70 g.

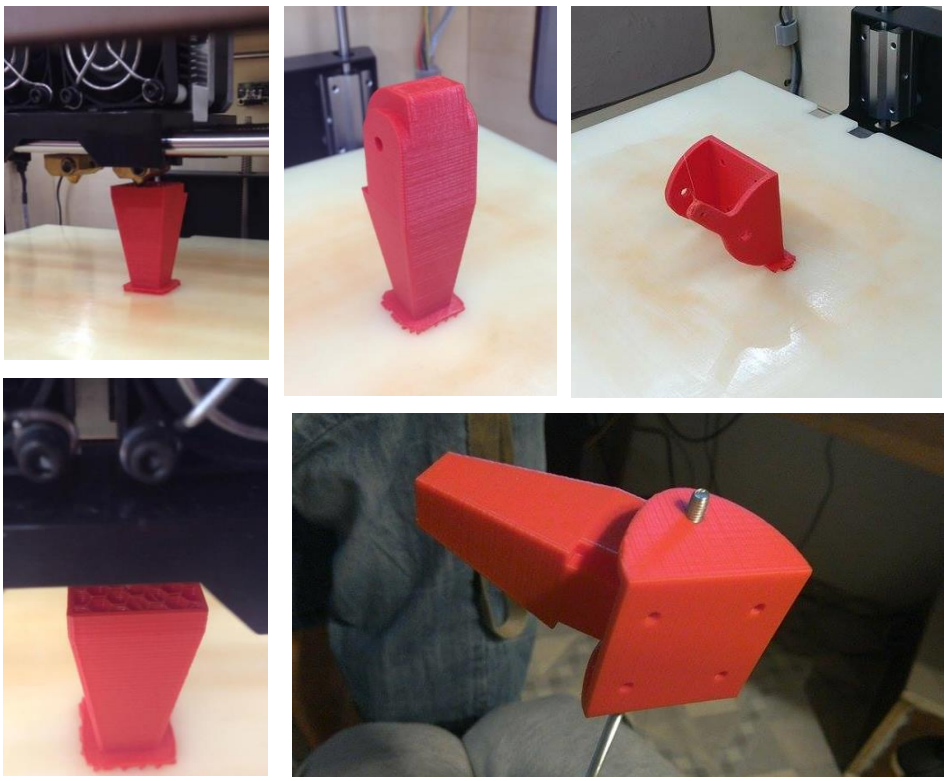


Obr. 8 – Modelování překlápěcího systému motorů

Zkouška zatížení držáků motorů:



Obr. 9 – Simulace deformací modelu při zatížení 10 kg



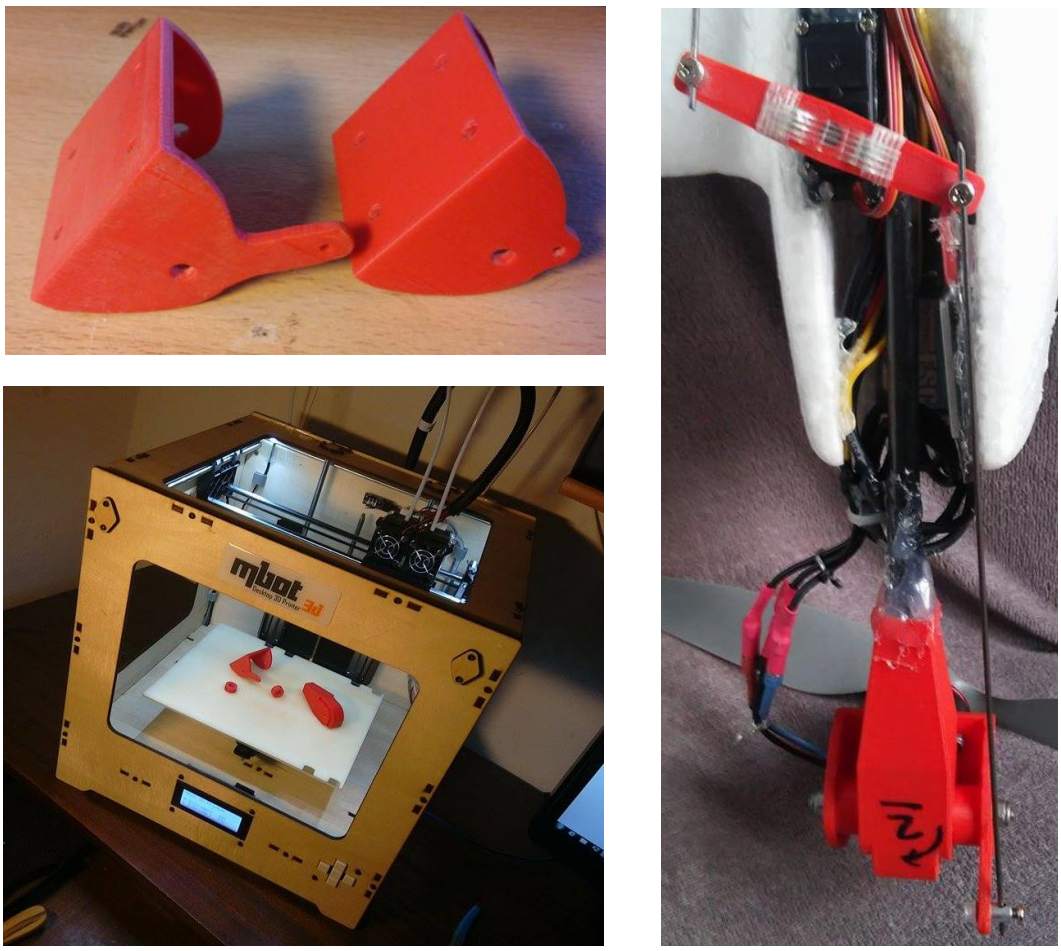
Obr. 10 - 3D tisk výsledného produktu 3D modelování

Další výhodou Inventoru byla možnost provedení silových testů na námi vymodelované sestavě. Ze simulace zatížení držáků motorů vyšel v programu lépe plast, neboť hliník vydržel sice větší zatížení a méně se deformoval, ale zato mnohonásobně namáhal ocelovou osičku překlápěcího systému. Při zátěžích, které na modelu při skutečném letu dosahujeme, postačilo ABS. Ukázky simulace zatížení na obrázcích níže jsou pouze ilustrační a je na nich viditelně zobrazena deformace držáků. Oproti realitě je zvolena mnohem větší zátěž, než můžeme na modelu dosáhnout, tedy 10 kg.

3.2 3D tisk

3D tisk je proces, při kterém se z digitální předlohy – 3D modelu vytváří fyzický model. Je to proces aditivní, to znamená, že se materiál přidává. Na rozdíl od obráběcích strojů, kde se materiál z celistvého bloku odebírá, až zbude jen požadovaný tvar. Výsledné 3D modely překlápěcího systému z prostředí Inventor jsme v tomto CAD prostředí převedli do mesh neboli mřížkového 3D modelu s příponou .stl, který je schopen zpracovávat software od 3D tiskárny. Ten pak následně automaticky, dle námi zadaných parametrů, vytvořil program, podle něhož 3D tiskárna model tiskla (obr. 10, 11). Pro náš překlápěcí systém jsme nastavili tisk se vnitřní strukturou v podobě včelích pláství, čímž bylo zajištěno, aby byl model lehký. Oproti původně vypočítaným 70 g váží pouze 34 g a i při této hmotnosti si zachovává svou pevnost, dokonce má větší pružnost. Tisk jednoho celého překlápěcího systému trval v součtu přibližně dvě a půl hodiny.

Pro překlápěcí část držáku motoru jsme zkoušeli dvě možnosti velikosti páky překlápění. Nakonec testováním vyšla lépe část s delší pákou, protože u ní tolik nezáleželo na přesnosti otočení serva a zároveň se lépe tiskla i páka k servu, u níž jsme pro jistotu zvolili poměr 1:1,2.



Obr. 11 – 3D tisk překlápěcího systému a montáž

3.3 Sestavení letadla

Při sestavování letadla jsme využili naše několikaleté modelářské zkušenosti. Konstrukce, která drží motory, je z uhlíkových trubek a kříží se s hlavním nosníkem křídla (obr. 12). Díky tomu je celá konstrukce opravdu velice tuhá, což je potřeba pro mód quadrocoptery. Na uhlíky jsme poté spolu s motory umístili držáky motorů, které jsme předtím vytiskli na 3D tiskárně. Zároveň jsme zachovali původní rozbíhavost a stoupavost motorů ve vodorovné poloze, které jsou zapotřebí pro stabilitu letadla.

3.4 Zapojení elektroniky

K rozvedení elektroniky mezi trupem a křídly jsme využili vnitřní konstrukci křídel, díky které jsme mohli umístit kabely dovnitř křídla (obr. 13). Gondoly, které původně sloužily k uchycení motorů, jsme využili pro přilepení regulátorů motorů a umístění serv překlápěcího systému motorů. Díky tvaru gondol se regulátory při letu zároveň chladí proudícím vzduchem.

4. Blokové schéma zapojení řízení (obr. 14)

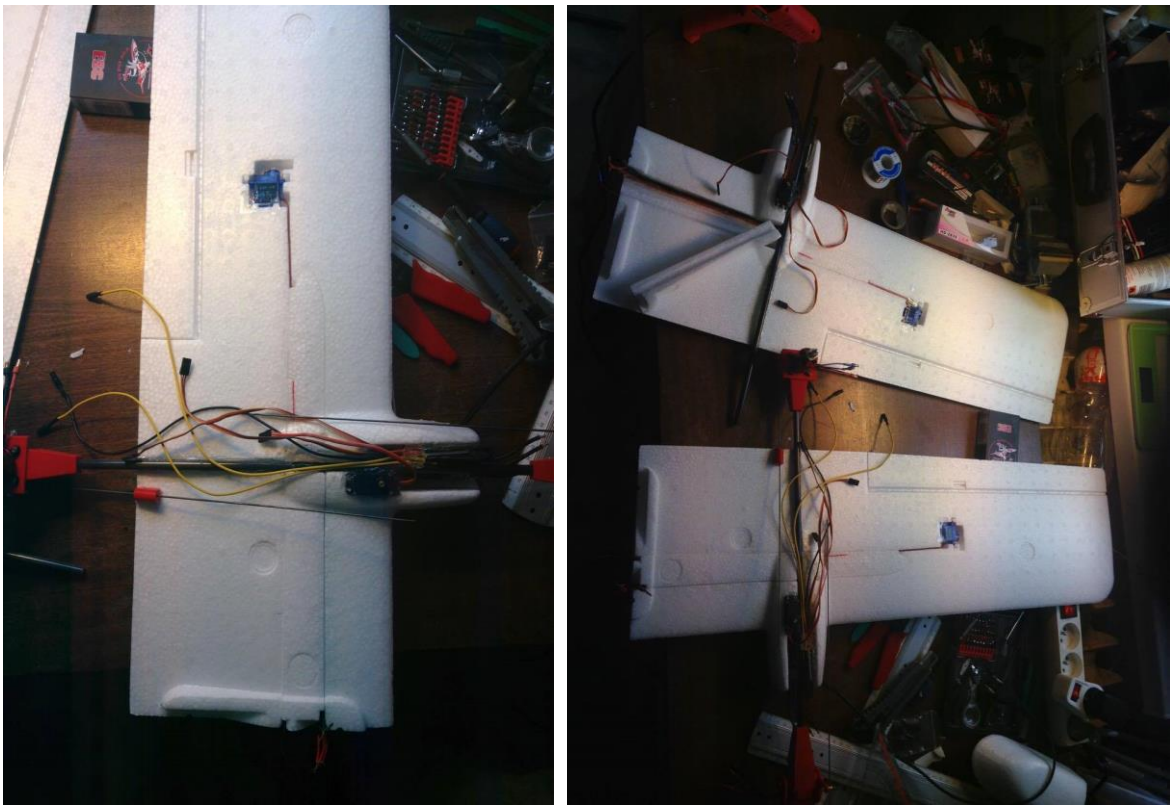
Model letadla je řízen pomocí dvou řídících jednotek Arduina a KK2. Obě jednotky zpracovávají pokyny pilota, který model ovládá rádiem ze země. Arduino programově řídí postupné překlopení motorů o 90° a posílá jejich polohu do KK2.0, která podle těchto signálů určuje svůj letový režim. KK2.0 tedy řídí celý model a podle letového režimu postupně

snižuje vliv gyroskopů při přechodu z visu na horizontální let. Zároveň přerozděluje funkce přijímaných kanálů pro veškeré další elektronické vybavení letadla.

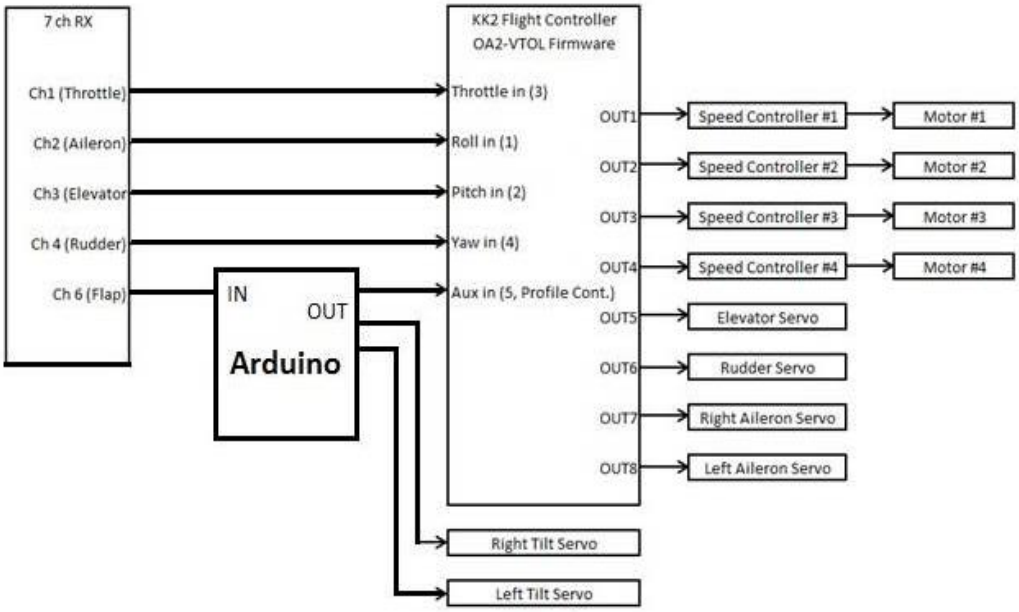
Obr. 12 – Montáž uhlíkových trubek pro uchycení motorů

4.1 Zapojení FPV

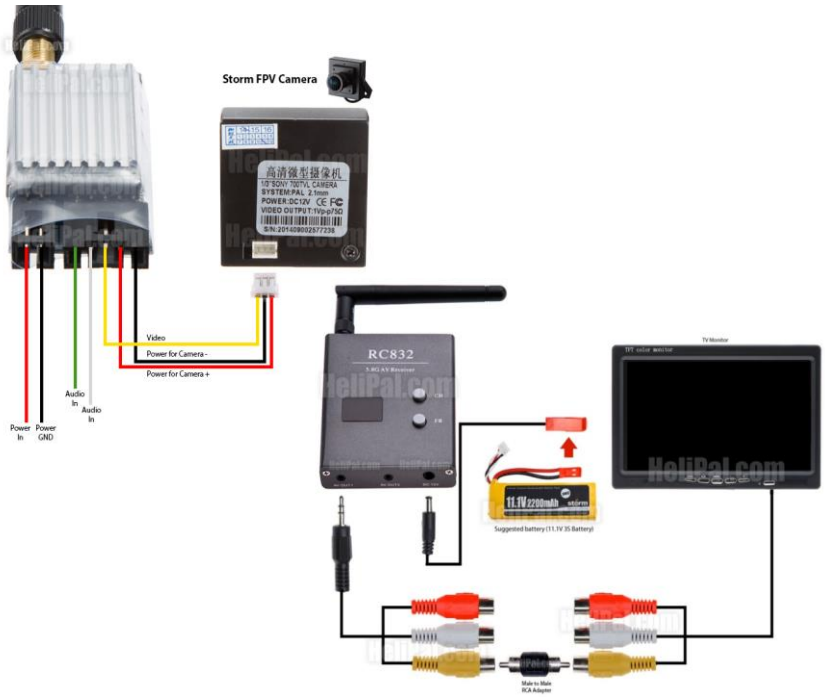
FPV systém je nezávislý na řídicím systému. V České republice bohužel z finančních důvodů nešlo zakoupit OSD (On Screen Display), který se připojuje k FPV a slouží k zobrazování letových údajů - rychlost, vzdálenost od místa startu, výška atd. - na displeji pilota. OSD umí také kooperovat s řídicím systémem. Náš FPV systém tedy slouží pouze k ukázce existence tohoto systému a pozorovacím účelům. Není určen k přímému řízení letadla podle video přenosu. To by v dnešní době bez dalších povolení nebylo v souladu s legislativou České republiky.



Obr. 13 – Osazení elektroniky a kabelů na křídlech



Obr. 14 – Blokové schéma zapojení řízení



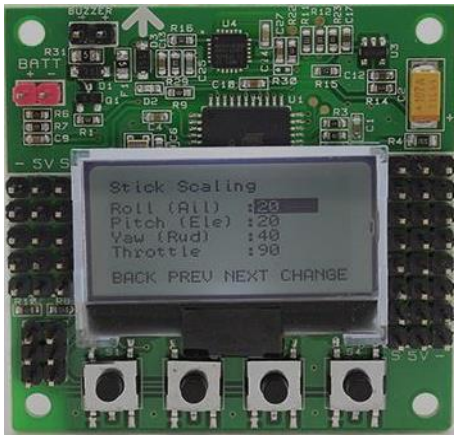
Obr. 15 – Zapojení FPV

5. Řídící deska a programování

5.1 Řídící jednotka KK2.0

Pro řízení ovládacích prvků modelu jsme použili běžně dostupnou řídící desku pro coptery s názvem KK2.0, která podle svých interních gyroskopů řídí a stabilizuje celé letadlo. Tato deska je založena na čipu Atmega32u4, se kterým jsme díky Arduinu již měli dříve

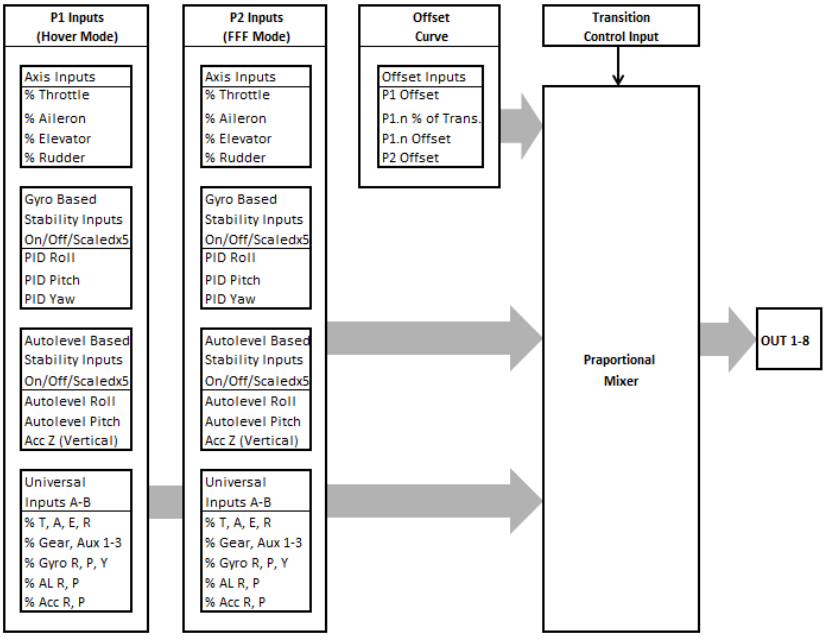
zkušenosti, takže se nám s deskou dobře pracovalo. KK2.0 má také integrovaný displej a tlačítka, na kterých se po nahrání hlavního softwaru doprogramovávají další funkce a individuální nastavení modelu.



Obr. 16 – Řídící jednotka

5.2 Program KK2.0

Jako řídící software pro KK2.0 se nám podařilo najít open source program OpenAeroVTOL, takže jsme mohli upustit od původního plánu, kdy jsme chtěli KK2.0 využívat pouze jako jednotku pro řízení v módu quadrocoptery a přechod mezi režimy jsme chtěli řešit pomocí Arduina. Díky využití programu OpenAeroVTOL zůstala kompletní stabilizace a změna letových režimů na KK2.0.
Program OpenAeroVTOL je vlastně základní program řídící desky KK2.0, jen v něm nejsou předdefinované funkce pro modely (jako například quadrocoptera). Tyto funkce jsme v našem případě programovali pro jednotlivé výstupy přímo na displeji KK2.0. Například v módu P1 – v našem případě quadrocoptera – bude na výstupu 1 regulátor pravého předního motoru, který bude v tomto módu reagovat na gyroskopy dle nastavené PID regulace a z 30 % bude ovládán řídícími signály výškovky, směrovky a křidélek. Ze 100 % bude reagovat na plyn.

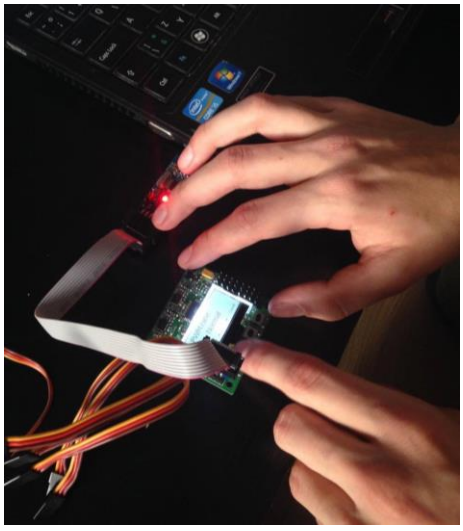


Obr. 17 – Blokové schéma funkce programu v KK 2.0

OpenAeroVTOL se tedy dá využít i na jiné účely než je kombinace coptery a letadla. Našemu modelu vyhovoval tím, že je schopen přecházet mezi jednotlivými režimy P1 a P2 pozvolna, podle předem vybraného průběhu. Kompletní nastavení KK2.0 pro náš model a větší blokové schéma funkce programu je v příloze.

Tabulka č. 1 – Nastavení PI regulace gyroskopů v KK2.0

Flightprofiles	**7. Profile1**	**8. Profile 2**
Roll P:	75	0
Roll I:	15	0
Roll I Limit:	10	0
Roll I Rate:	1	1
RollAutoLvl:	8	0
RollTrim:	0	0
Pitch P:	75	0
Pitch I:	15	0
Pitch I Limit:	15	0
Pitch I Rate:	1	1
PitchAutoLvl:	8	0
PitchTrim:	0	0
Yaw P:	120	30
Yaw I:	0	0
Yaw I Limit:	0	0
Yaw I Rate:	2	1
YawTrim:	0	0
AccVert P:	40	0

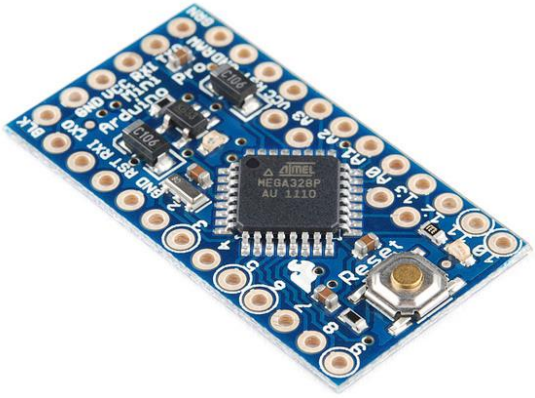


Obr. 18 – Ruční programování přímo na řídící jednotce

Tabulka č. 2 – Nastavení výstupů a jejich funkce v KK2.0

9.-16. OUT1-8 Mixers

Function	#1 Motor	#2 Motor	#3 Motor	#4 Motor	Elevator	Rudder	RightAil	LeftAil
Settingfor:	OUT1	OUT2	OUT3	OUT4	OUT5	OUT6	OUT7	OUT8
Device:	Motor	Motor	Motor	Motor	Servo	Servo	Servo	Servo
P1 Offset:	0	0	0	0	0	0	0	0
P1.n % of trans:	15	15	15	15	15	15	15	15
P1.n Offset:	-2	-2	2	2	0	0	0	0
P2 Offset:	0	0	0	0	0	0	0	0
P1 Thr. volume:	100	100	100	100	0	0	0	0
P2 Thr. volume:	100	100	100	100	0	0	0	0
Throttlecurve	linear	linear	linear	linear	linear	linear	linear	linear
P1 Ail. volume	25	-25	-25	25	0	0	-80	-80
P2 Ail. volume	0	0	0	0	0	0	-80	-80
P1 Ele. volume	-25	-25	25	25	-80	0	0	0
P2 Ele. volume	0	0	0	0	-80	0	0	0
P1 Rud. volume	-35	35	-35	35	0	80	0	0
P2 Rud. volume	10	-10	-10	10	0	80	0	0
P1 RollGyro:	ON	ON	ON	ON	OFF	OFF	ON	ON
P2 RollGyro:	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
P1 PitchGyro:	ON	ON	ON	ON	OFF	OFF	OFF	OFF
P2 PitchGyro:	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
P1 YawGyro:	ON	ON	ON	ON	OFF	OFF	OFF	OFF
P2 YawGyro:	ON	ON	ON	ON	OFF	OFF	OFF	OFF
P1 Roll AL:	ON	ON	ON	ON	OFF	OFF	ON	ON
P2 Roll AL:	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
P1 Pitch AL:	ON	ON	ON	ON	OFF	OFF	OFF	OFF
P2 Pitch AL:	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
P1 Z acc:	ON	ON	ON	ON	OFF	OFF	OFF	OFF
P2 Z acc:	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
P1 Source A:	AccRoll	AccRoll	AccRoll	AccRoll	None	AccRoll	None	None
P1 Volume:	30	-30	30	-30	0	-25	0	0
P2 Source A:	AccRoll	AccRoll	AccRoll	AccRoll	None	AccRoll	None	None
P2 Volume:	-20	20	20	-20	0	-25	0	0
P1 Source B:	None	None	None	None	None	None	None	None
P1 Volume:	0	0	0	0	0	0	0	0
P2 Source B:	None	None	None	None	None	None	None	None
P2 Volume:	0	0	0	0	0	0	0	0



Obr. 19

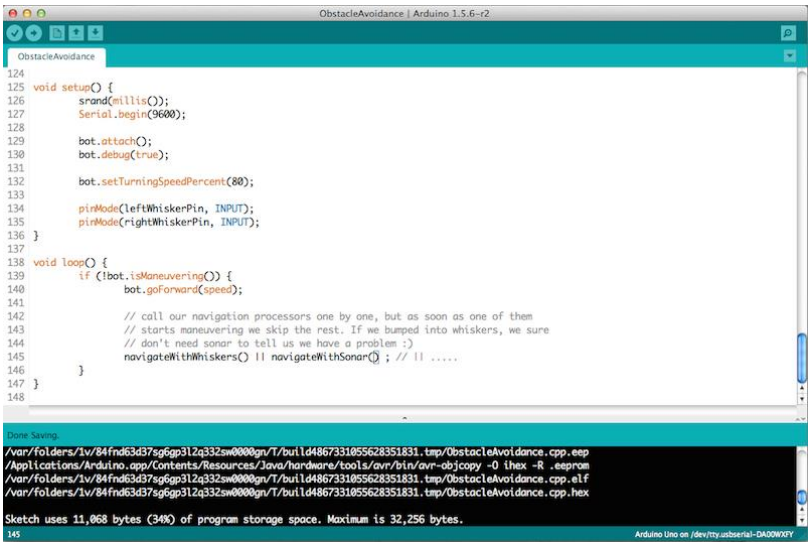
5.3 Řídící deska Arduino

U našeho Quadroletadla jsme také použili řídící desku Arduino mini (obr. 19), která je založená na procesoru Atmega 328 a má 13 digitálních vstupně-výstupních pinů (z toho 6 s podporou PWM) a 6 analogových vstupů. Našemu projektu vyhovuje díky svým malým rozměrům.

Arduino mini se stará o plně automatické postupné překlápění motorů a odesílá informace o poloze motorů do desky KK2.0 přes servosignály.

5.3.1 Programování Arduino

Arduino se programuje pomocí speciálního Arduino programovacího jazyka (založeného na jazyku Wiring – vycházející z C++) ve vlastním Arduino vývojovém prostředí. Projekty založené na Arduinu mohou jednoduše komunikovat se softwarem na stolním počítači nebo notebooku.



Obr. 20 – Programovací prostřední Arduina

5.3.2 Arduino programy

Pro naše Arduino jsme vytvořili několik programů:
a) Nastavení maximálních hodnot serv pomocí potenciometrů a výpisu hodnot na PC.
Jednoduchý program pro prvotní zjištění maximálních poloh překlápěcích serv a překlápění u KK2.0. Serva se překlápěla pomocí čtení napěťové hodnoty na potenciometru a odesílala tuto hodnotu na displej PC.

Zjistili jsme hodnoty serv pro vodorovný let a pro překlopení motorů o 90° (mód quadrocoptery).

Program a)

```
#include<Servo.h>
Servomyservo;  // vytvoření objektu servo pro kontrolu serva
intpotpin = 0;  // analog pin použitý na potenciometr
int val;    // proměná pro analog pin
voidsetup() {
Serial.begin(9600); // spustí komunikaci s pc přes rs-232 (com)
myservo.attach(9);  // servo je zapojené na pin 9 na arduinu
}
voidloop() {
  val = analogRead(potpin);           // přečte hodnotu potenciometru (hodnota mezi 0 a 1023)
  val = map(val, 0, 1023, 0, 180);     // podle měřítka vypočítá hodnotu (hodnota mezi  0 a 180)
myservo.write(val);                  // zapíše hodnotu proměnné na servo
Serial.println(val);                // odešle hodnotu serva do PC
delay(50);                          // čekání 50ms než servo dorazí na tuto hodnotu
}
```

b) Třípolohové překlopení pomocí signálů z přijímače.
Pomocí třípolohového přepínače na vysílači jsme přepínali mezi módy letadla, přechodu a quadrocoptery.
Tento program jsme použili na první testování modelu a pro test, zda námi navržený systém překlápění bude schopen překlopit i běžící motory.

Program b)

```
#include<Servo.h>
Servomyservo;
Servo myservo2;
Servo kk2;
constintchA=3;      // pin na kterém je výstup z přijímače

int ch1;
int a;
int b;

constintmn = 8; // minimální poloha serva -> 90°
constint st = 40; // střední přechodová poloha
constintmx = 85; // maximální poloha -> 0°

constint mn2 = 112; // minimální poloha serva -> 90°
constint st2 = 80; // střední přechodová poloha
```

```
constint mx2 = 39; // maximální poloha -> 0°

constint mn3 = 0; // minimální poloha serva -> 90°
constint st3 = 90; // střední přechodová poloha
constint mx3 = 180; // maximální poloha -> 0°
voidsetup() {

pinMode(chA, INPUT);

myservo.attach(5); // 10->90 červená páka
  myservo2.attach(6); // ovázaný kabel 110 -> 40 bílá páka
  kk2.attach(9);

  a=mn;
  b=mn2;
}

voidloop() {
  ch1 = pulseIn (chA,HIGH); //
if (ch1 < 1250){
myservo.write(mn);
    myservo2.write(mn2);
    kk2.write(mn3);
  }
if ((ch1 >1250)  && (ch1< 1650)){
myservo.write(st);
    myservo2.write(st2);
    kk2.write(st3);
  }
if (ch1 > 1650){
myservo.write(mx);
    myservo2.write(mx2);
    kk2.write(mx3);
  }
}
```

c) Pomalé plynulé překlápění motorů.

Tento program jsme vytvořili kvůli prezentaci modelu, aby zde bylo jasně patrné postupné překlápění motoru do určitých poloh.

Program proto stále reaguje na třípolohový přepínač a pouze plynule přechází do požadované polohy.

```
#include<Servo.h>

Servomyservo;
Servo myservo2;
Servo kk2;

constintchA=3;  // pin na kterém je výstup z přijímače
```

```
int ch1;

int a;
int b;
int c;

constintmn = 12; // minimální poloha serva -> 90°
constint st = 40; // střední přechodová poloha 38.5 // od 8 32 skoků do 40
constintmx = 90; // maximální poloha -> 0°
//77 skoků červená

constint mn2 = 112; // minimální poloha serva -> 90°
constint st2 = 88; // střední přechodová poloha 36.5 //112-32=80
constint mx2 = 35; // maximální poloha -> 0°
//73 skoků bílá

constint mn3 = 0; // minimální poloha serva -> 90°
constint st3 = 90; // střední přechodová poloha
constint mx3 = 180; // maximální poloha -> 0°
// 180 skoků -> 2.5 až 2.4 krát pomalejší

voidsetup() {

pinMode(chA, INPUT); // signály z přijímače vstup

myservo.attach(5); // 10->90 červená páka
  myservo2.attach(6); // ovázaný kabel 110 -> 40 bílá páka
  kk2.attach(9);

  a=st;
  b=st2;
  c=st3;

 ch1 = pulseIn (chA,HIGH);
if (ch1 < 1250){
  a=mn;
  b=mn2;
  c=mn3;
  }
if ((ch1 >1250)  && (ch1< 1650)){
  a=st;
  b=st2;
  c=st3;
  }
if (ch1 > 1650){
  a=mx;
  b=mx2;
  c=mx3;
  }
```

```
}

voidloop() {
  ch1 = pulseIn (chA,HIGH); //
if (ch1 < 1250){
if (a >mn){ //
myservo.write(a);
    a=a-1;}
if (b < mn2){
    myservo2.write(b);
    b=b+1;}
if (c > mn3){
    kk2.write(c);
    c=c-2;}
if (a <= mn)
myservo.write(mn);
if (b >= mn2)
    myservo2.write(mn2);
if (c <= mn3)
    kk2.write(mn3);
    }
if ((ch1 >1250)  && (ch1< 1650)){
if (a > st){ //a větší než st=40 -1
myservo.write(a);
    a=a-1;
    }
if (a < st){ //a menší st=40 přičti +1
myservo.write(a);
    a=a+1;
    }
if (b < st2){
    myservo2.write(b);
    b=b+1;
    }
if (b > st2){
    myservo2.write(b);
    b=b-1;
    }
if (c > st3){
    kk2.write(c);
    c=c-2;}
if (c < st3){
    kk2.write(c);
    c=c+2;}
if (a == st)
myservo.write(st);
if (b == st2)
    myservo2.write(st2);
if (c == st3)
```

```
    kk2.write(st3);
  }
if (ch1 > 1650){
if (a <mx){
myservo.write(a);
    a=a+1;}
if (b > mx2){
    myservo2.write(b);
    b=b-1;}
if (c < mx3){
    kk2.write(c);
    c=c+2;}
if (a >= mx)
myservo.write(mx);
if (b <= mx2)
    myservo2.write(mx2);
if (c >= mx3)
    kk2.write(mx3);
  }
delay(30);  //rychlost přechodu
}
```

d) Letové plynulé překlápění motorů.
Tento program automaticky přesouvá motory do určitých poloh v závislosti na čase.

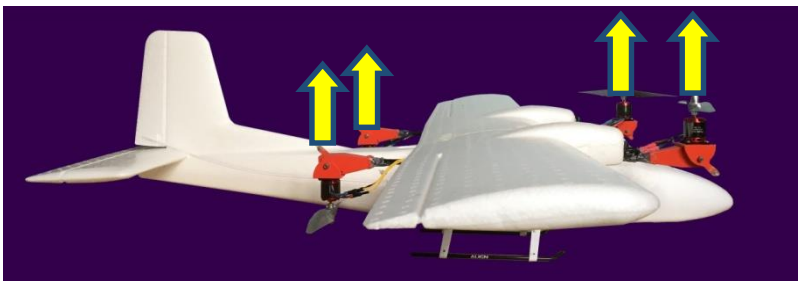
Program reaguje pouze na dvoupolohový přepínač a plynule se přesouvá z módu quadrocoptery do módu přechodu zde setrvá 500 ms a poté otáčí servy dále do módu letadla. Po přepnutí zpět do módu quadrocoptery plynule překlopí motory o 90°.

```
#include<Servo.h>
Servomyservo;
Servo myservo2;
Servo kk2;

constintchA=3;  // pin na kterém je výstup z přijímače

int ch1;

int a;
int b;
int c;

constintmn = 12; // minimální poloha serva -> 90°
constint st = 40; // střední přechodová poloha 38.5 // od 8 32 skoků do 40
constintmx = 90; // maximální poloha -> 0°
//77 skoků červená

constint mn2 = 112; // minimální poloha serva -> 90°
constint st2 = 88; // střední přechodová poloha 36.5 //112-32=80
constint mx2 = 35; // maximální poloha -> 0°
```

```
//73 skoků bílá

constint mn3 = 0; // minimální poloha serva -> 90°
constint st3 = 90; // střední přechodová poloha
constint mx3 = 180; // maximální poloha -> 0°
// 180 skoků -> 2.5 až 2.4 krát pomalejší

voidsetup() {

pinMode(chA, INPUT); // signály z přijímače vstup

myservo.attach(5); // 10->90 červená páka
  myservo2.attach(6); // ovázaný kabel 110 -> 40 bílá páka
  kk2.attach(9);

  a=st;
  b=st2;
  c=st3;

 ch1 = pulseIn (chA,HIGH);

if (ch1 < 1250){
  a=mn;
  b=mn2;
  c=mn3;
  }
if (ch1 > 1650){
  a=mx;
  b=mx2;
  c=mx3;
  }

}

voidloop() {

  ch1 = pulseIn (chA,HIGH);

if (ch1 < 1250){
if (a >mn){
myservo.write(a);
    a=a-5;}
if (b < mn2){
    myservo2.write(b);
    b=b+5;}
if (c > mn3){
    kk2.write(c);
    c=c-12;}
if (a <= mn)
myservo.write(mn);
```

```
if (b >= mn2)
   myservo2.write(mn2);
if (c <= mn3)
   kk2.write(mn3);
   }

if (ch1 > 1650){

if (a <mx){
myservo.write(a);
   a=a+5;}

if (b > mx2){
   myservo2.write(b);
   b=b-5;}

if (c < mx3){
   kk2.write(c);
   c=c+12;}

if (a >= mx)
myservo.write(mx);
if (b <= mx2)
   myservo2.write(mx2);
if (c >= mx3)
   kk2.write(mx3);

if ((c > st3 - 10) && (c < st3 + 10)){ // čekání v střední poloze
   kk2.write(st3);
myservo.write(st);
   myservo2.write(st2);
delay(500);
   }
  }
delay(25);     // rychlé časování
}
```

6. Překlápění motorů a funkce řídících jednotek

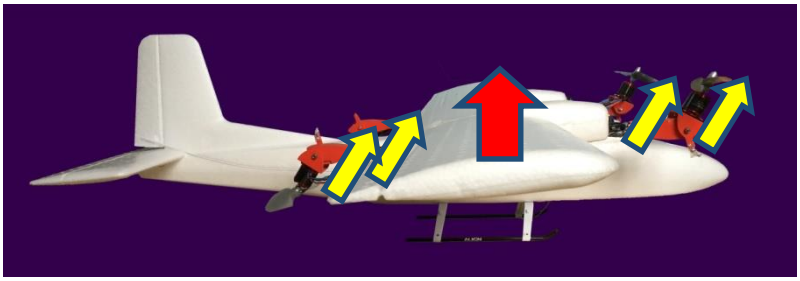


Mód quadrokoptéry
-vztlak a stabilitu zajišťuje dělení Výkonu na motorech pomocí řídící jednotky KK2.0.

Obr. 21

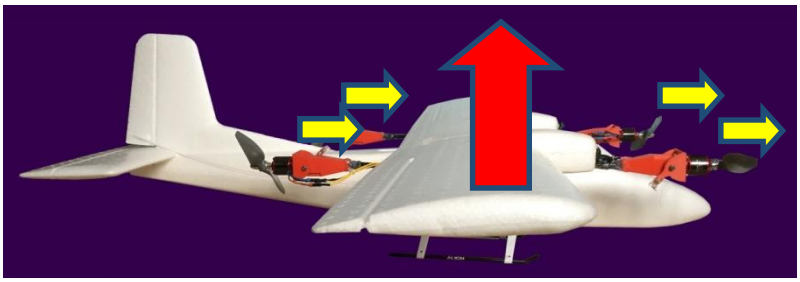
První letový režim je mód quadrocoptery: v tomto režimu nesou celou tíhu modelu pouze motory. Řídící jednotka pak rozdělováním výkonu mezi tyto motory zajišťuje stabilitu letadla a pohyb letadla.

Druhý letový režim je mezipoloha, ve které se postupným přesměrováním tahů motorů ze svislé do vodorovné polohy letadlo rozlétá. Díky zvyšování rychlosti letadla stoupá vztlaková síla křídla, která kompenzuje postupnou ztrátu svislého tahu motorů.



Obr. 22

Posledním letovým režimem je mód letadla, ve kterém vztlaková síla křídla, která vznikla díky rychlosti letadla, nese celou tíhu modelu.



Obr. 23

7. Další možnosti rozšíření

Do budoucna bychom rádi nahradili překlápění pomocí času, překlápěním pomocí reálné rychlosti letadla, což by bylo zjišťováno Pitotovou trubicí. Bohužel samotná Pitotova trubice není v České republice běžně dostupná.

8. Závěr

Prostřednictvím řídících jednotek bylo dosaženo plynulého přechodu z módu quadrocoptery na mód letadla. Tím byla docílena úspora energie při horizontálním letu a snížena potřeba prostoru při vzletu a přistání. Další výhodou tohoto modelu je, že může doletět na určené místo, kde se přepne do módu quadrocoptery. Zde může nehybně setrvávat ve visu nad konkrétním místem a monitorovat situaci na zemi bez nutnosti kroužení.

9. Použité webové stránky

http://www.rcgroups.com/forums/showthread.php?t=1814667
http://www.rcgroups.com/forums/showthread.php?t=1972686