

**Gymnázium, Praha 6, Arabská 14**

Programování



**ROČNÍKOVÝ PROJEKT**

**Bezpilotní letadlo**

Vypracovali:

Vedoucí práce:

Havránek Kryštof  
ing. Daniel Kahoun

Únor 2022

Prohlašujeme, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská 14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V ..... dne .....

Podpis autora

Název práce: Bezpilotní letadlo

Autor: Havránek Kryštof

Abstrakt:

Klíčová slova: (UAV), (Gtk), (C++), (Raspberry zero 2)

---

Title: Drone

Author: Havránek Kryštof

Abstract:

Key words: (UAV), (Gtk), (C++), (Raspberry zero 2)

---

Title: Drohne

Autor: Havránek Kryštof

Abstrakt:

Schlüsselwörter: (UAV), (Gtk), (C++), (Raspberry zero 2)

# Obsah

<b>Úvod</b>	<b>iv</b>
<b>1 Rozložení práce</b>	<b>1</b>
1.1 Komunikační protokol . . . . .	1
1.1.1 Struktura protokolu . . . . .	1
1.1.2 Specifikace protokolu . . . . .	2
<b>2 Hardware a Raspberry Pi</b>	<b>4</b>
2.1 Program . . . . .	7
2.2 Interface s Hardwarem . . . . .	7
2.3 Organizace v kódu . . . . .	8
2.4 Ovládání letadla . . . . .	9
2.4.1 Autopilot . . . . .	9
<b>3 Client na ovládání</b>	<b>10</b>
<b>Závěr</b>	<b>vi</b>
<b>Seznam použité literatury</b>	<b>vii</b>
<b>Seznam obrázků</b>	<b>viii</b>

# Úvod

# 1. Rozložení práce

Práci lze rozdělit na dva základní komponenty. První část software běží na počítači přes který se bezpilotní letadlo ovládá, druhá na samotném dronu. Obě části jsou psané v jazyce C++.

Software pro počítač používá grafickou knihovnu Gtk3 a byl vyvíjen primárně na operačním systém Linux. Gtk3 je multiplatformní toolkit, port na další operační systémy je tak možný a kód je psán stylem, aby umožnil další verze.

Jádro samotného dronu představuje malý počítač Raspberry Pi Zero 2. Vyšší výkon verze Zero 2 není nutný pro fungování, práce tak defacto může fungovat na libovolném Raspberry Pi. Program využívá jen zlomek zdrojů arduino a to jak po stránce paměti, tak výpočetního výkonu.

Obě části mezi sebou komunikují prostřednictvím protitoku založené na TCP rodině. Přenos videa je zprostředkovávám pomocí UDP. Samotné video je kódováno do jpg, v porovnání s dalšími styly přenosu byla latence u jpg menší.

## 1.1 Komunikační protokol

Ovládací počítač a Raspberry mezi sebou komunikují prostřednictvím jednoduchého TCP protokolu. Raspberry Pi přebýrá na sebe roli serveru, to mimo jiné umožňuje aby bylo zařízení ovládáno z více stanic. Případně alespoň telemetrie byla posílána na více zařízení.

Protokol prozatím není zašifrovaný a nepočítá s nevalidními daty. Šifrován není příliš nutné, dronu a počítač operují na vlastní síti, kde Raspberry Pi funguje jako Wi-Fi Access–Point. Struktura protokolu ovšem nabízí možnost jednoduše naimplementovat symetrickou kryptografií.

### 1.1.1 Struktura protokolu

Každý packet začíná hlavičkou o velikosti 5 bytů.

1. typ zprávy
2. priorita – Program prozatím prioritu nebere v potaz.

3. 8 horních bitů velikosti zprávy
4. 8 dolních bitů velikosti zprávy
5. dopočet do kontrolního součtu – součet čísel v hlavničce musí být dělitelný sedmi.

Po hlavičce následuje samotná zpráva zakončená posloupností pěti bytů – 0x00, 0x00, 0xFF, 0xFF, 0xFF. Zakončovací řetězec slouží pro případ, kdy nějaký packet nedorazí, či příchozí zpráva byla nějak poškozena. Nastane li tato situace program postupně čte byty ze socketu a hledá tuto sekvenci. Maximální délka zprávy je půl kilobytu, samotná data jsou posílána v balíčcích o maximální velikosti 250 bytů. Omezení velikosti zprávy je zavedeno hlavně kvůli šetření paměti – program má pro každého klienta vytvořené jedno pole, do kterého se musí vejít celá zpráva.

Protokol funguje na základě posílání struktur, definovaných v `protocol_spec.h`. Pro odeslání dat je nejprve nutné nahrát data do struktury. Dále se z těchto dat vytvoří tzv. `SendingStructure`, které se odešle do thread pool. Pro přijetí se data načítají do bufleru spojeným s klientem, ze kterého se později zkopiují objektu `ProcessingStructure`. Příslušná část programu si pak data přes `memcpy` opět zkopiuje na příslušnou strukturu.

Pro příjem a zpracování zpráv slouží dva thread pooly – `SendingThreadPool` a `ProcessingThreadPool`. Jednotlivé části programu tak nemusí čekat až se uvolní socket aby se mohla data odeslat, či čekat na jejich zpracování pro přijetí. Speciální případem je zpracování příkazů, které přímo ovládají pohyb letadla. Pro ovládaní existuje samostatné vlákno, které příkazy vykonává sekvenčně. Také je implementován mechanismus, kde příliš staré příkazy se nevykonají.

Program sleduje aktivitu na socket pomocí příkazu `select`. Při běžné provozu tak většina vláken komunikací spí a zátěž není příliš velká.

z hlediska programové stránky jsou veškeré součásti komunikace implementovány jako singletony. Program je tak řešen, aby umožnil jednoduše odesílat zprávy z různých částí programu.

### 1.1.2 Specifikace protokolu

Protokol obsluhuje tři základní části provozu dronu – nastavení, ovládaní a telemetrie. Pro nastavení je vyhrazen rozsah od 0x00 po 0x20 a zahrnuje věci jako odpojení

klienta, nastavení kamery, či restart.

Další kategorie představuje ovládaní, pro které je vyhrazen prostor mezi 0x21 až 0x40. Projekt implementuje dva základní typy – standardní ovládaní (pohyb joysticku, atd.) a speciální. Standardní slouží pro prosté manuální ovládání dronu. Speciální například resetuje polohoměr, či zapíná autopilota.

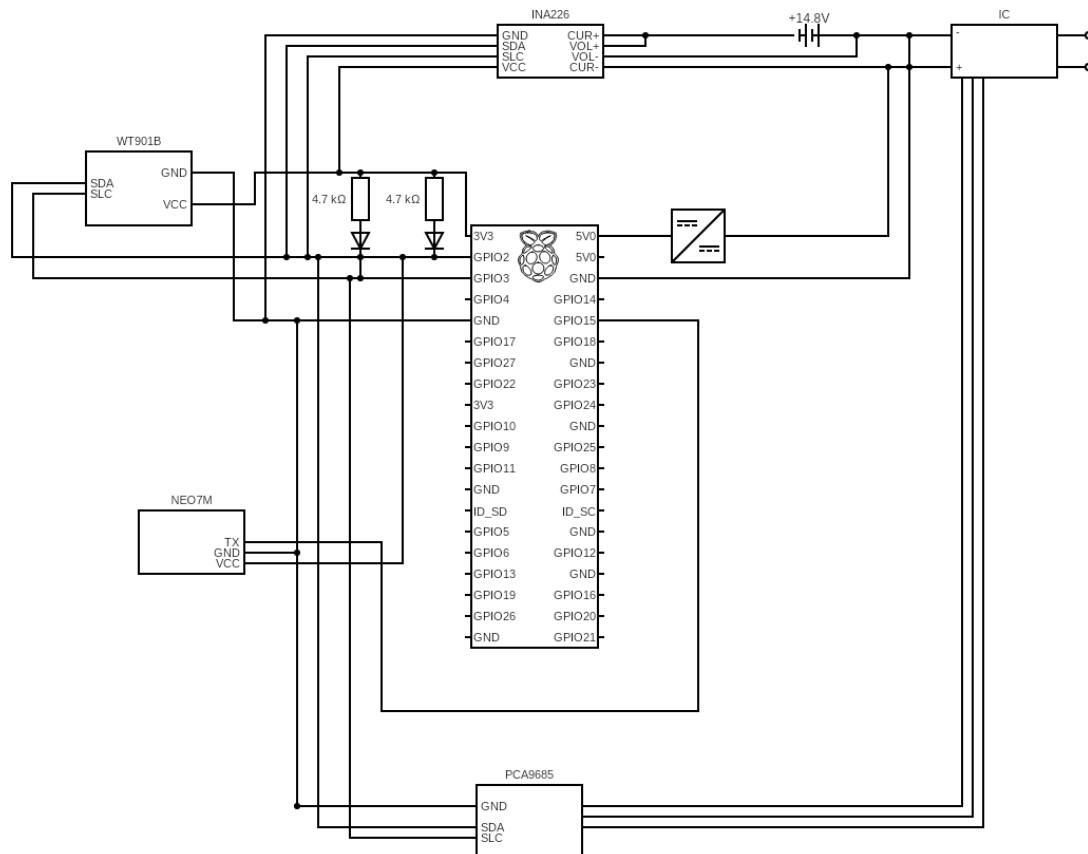
Poslední kategorií je telemetrie, které má zbytek rozsah do 0xFF. Tu ve standardním chování posílá Raspberry každých pár stovek milisekund. Chce-li client aktuálnější data může si je vyžádat odesláním zprávy o stejném typu, jako jsou požadovaná data. Speciální případ představují chybové hlášky, které vyžádat pochopitelně nejdou a pilotovi se zobrazují v dialogovém okně.

## 2. Hardware a Raspberry Pi

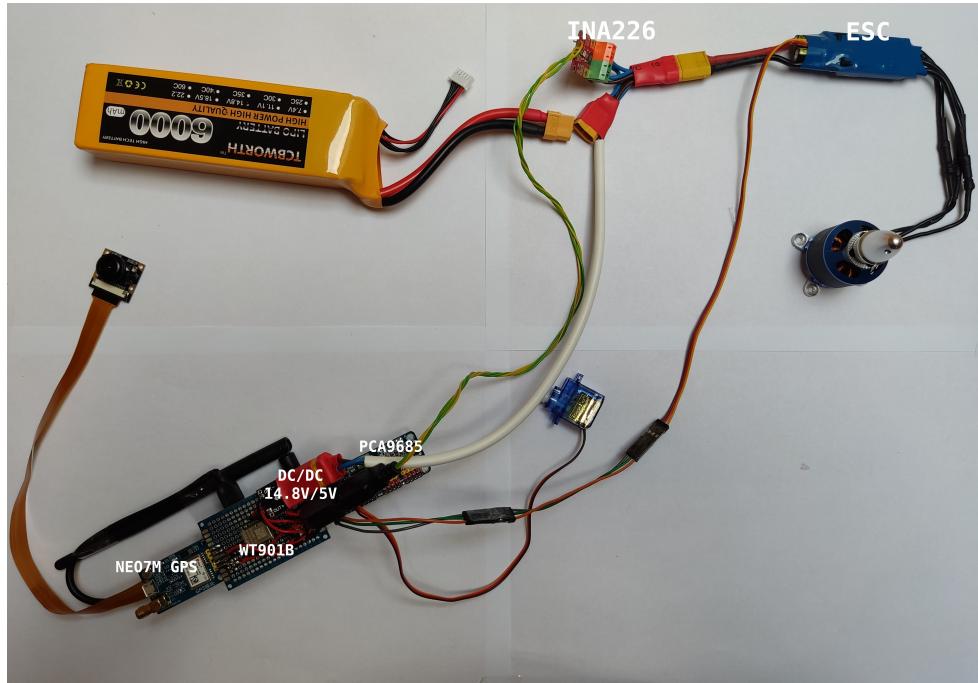
Jádro samotného letadla tvoří malý počítač Raspberry Pi Zero 2. Ostatní periférie jsou: polohový senzor WT901B, voltmetr a ampérmetr INA226, ovladač na serva PCA9865, ublox NEO-7M GPS modul a BEATLES 40A ESC. Vše s výjimkou GPS komunikuje s Raspberry Pi prostřednictvím I2C protokolu. GPS modul pak pomocí sériové linky.

Senzor WT901B sice podporuje přímé napojení GPS senzoru, při pokusu o konfiguraci senzoru však přestala fungovat sériová komunikace obecně. GPS je tak napojena sám.

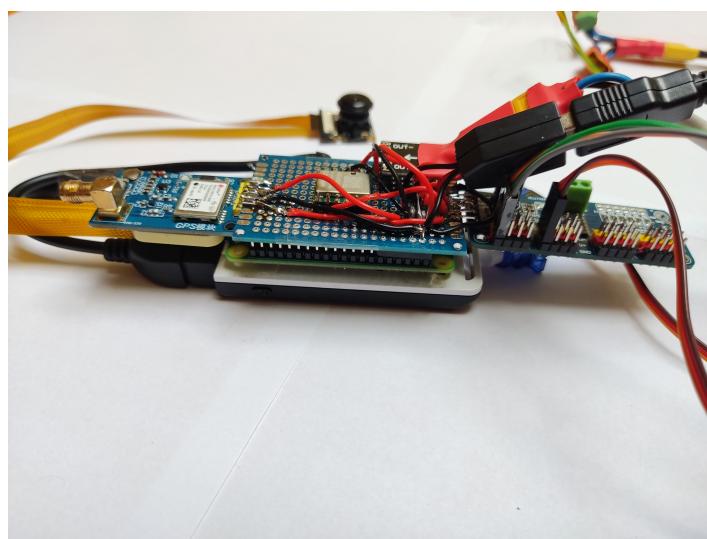
Systém má dva přívody proudu – hlavní motor a serva jsou napájeny prostřednictvím ESC. To však nebylo schopné poskytovat dostatečný proud pro Raspberry Pi a senzory na něj napojené. Z baterie je tak vyvedená linka napojená na step down converter, který 14.8 V sníží na 5V. Ty jsou přímo napojeny na Raspberry Pi.



Obrázek 2.1: Schéma zapojení obvodu

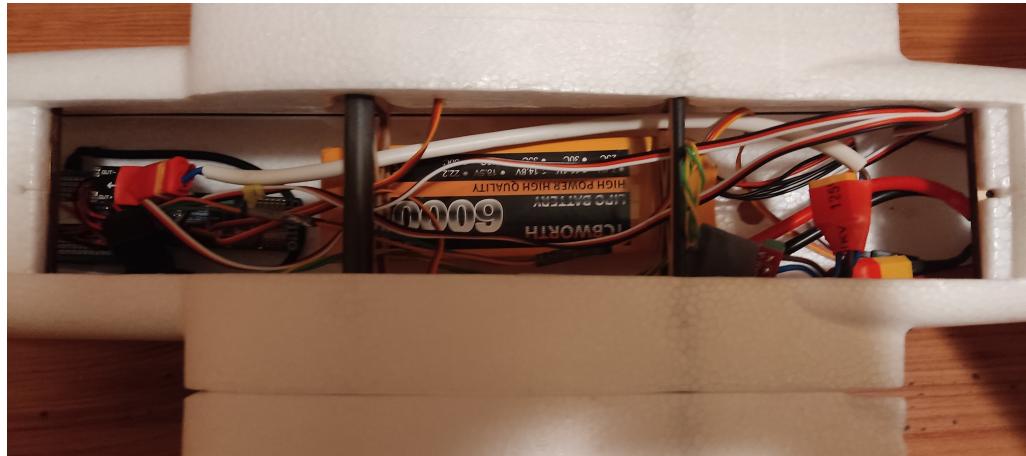
Obrázek 2.2: Reálné zapojení<sup>1</sup>

Většina komponentů je napájena na bread board, který lze jednoduše nasadit na Raspberry Pi. Výjimku představuje voltmetr/ampérmetr, který je volně a v letadle se nachází v ocase. Ke zbytku se připojuje přes koncovku USB, ostatní koncovky jsou XT60. V aktuální verzi je zapojení relativně nepraktické – z ocasu do přídě vedou tři kabely.



Obrázek 2.3: Detail Raspberry Pi

<sup>1</sup>TP-LINK WN772N je aktuálně volně vedle Raspberry Pi, nikoliv pod



Obrázek 2.4: Detail těla dronu



Obrázek 2.5: Celý dron

Kostroum dronu je model Mini Talon od čínské společnosti X-UAV. Rozpětí křídel je 130cm délka pak 85cm. Jedná se o tzv. V-Tail konfiguraci ocas tedy nemá standardní tři kontrolní plochy ale pouze dvě. V aktuální verzi program má naimplementované ovládání pouze pro tuto konfiguraci.

## 2.1 Program

```

drone_software
├── libraries
│   ├── Raspberry-JY901-Serial
│   ├── raspberry-pi-ina226
│   └── rpidmx512-Lib-PCA9685
├── include
│   ├── battery_interface.h
│   ├── camera_streamer.h
│   ├── communication_interface.h
│   ├── generic_PID.h
│   ├── gps_interface.h
│   ├── imu_interface.h
│   ├── protocol_spec.h
│   ├── servo_control.h
│   └── telemetry.h
└── src - implemetation of header files

```

Obrázek 2.6: Struktura souborů frontend

Program na Raspberry Pi používá tři základní knihovny pro komunikaci s WT901B, INA226 a PCA9685. Přenos z kamery pak zprostředkovává OpenCV, který jako backend používá gstreamer. V budoucnu je tak možné přidat zpracování videa přímo na raspberryPi – například detekce věcí na obrazu.

## 2.2 Interface s Hardwarem

Jak už bylo řečeno s hardwarem Raspberry komunikuje prostřednictví I2C spojení a sériové linky. Pro zpracování dat z INA226 a WT901B byly napsány nové knihovny. Jelikož knihovny pro Raspberry Pi totiž buďto neexistovaly, nebo nebyly zdaleka kompletní.

Knihovna pro WT901B vznikla forkem knihovny pro Arduino<sup>2</sup> a byla předělána v souladu s dokumentací protokolu<sup>3</sup>. Kvůli problémům se seznorem byla i dopsána I2C komunikace, kterou původní knihovna neimplementovala.

<sup>2</sup>TIAN, P. *Arduino-JY901-Serial*. GitHub. Dostupné z <https://github.com/paul-tian/Arduino-JY901-Serial>. [cit 2022-27-2]

<sup>3</sup>WT901 Datasheet. WitMotion Shenzen Co. Ltd. Dostupné z <https://www.wit-motion.com/gyroscope-module/Witmotion-wt901-ttl-i2c.html>. [cit 2022-27-2]

případě knihovny pro INA226 bylo forknuto demo<sup>4</sup> a dopsáno v plnohodnotnou knihovnu.

Obě knihovny využívají knihovnu wiringPi. Ta byla koncem roku 2021 spolu s vydáním raspbian bullseye označena ze deprecated. Také dochází ke kolizím mezi částmi kódu, které přistupují k I2C přes knihovnu WiringPi a částmi kódu, které přistupují přímo přes bcm2835. Nejedná se o kritickou chybu, ale někdy je narušena integrita některých dat.

Vzhledem k přímočarosti výstupu z GPS modulu je celá implementace přímo součástí hlavního kódu. Program zpracovává GPGGA packet, který obsahuje veškerá důležitá data. Zpracování dalších packetů nebylo pro práci podstatné.

V neposlední řadě k PCA9685 se také přistupuje přes knihovnu<sup>5</sup>. Knihovna je v základu psána pro Raspberry Pi a implementuje všechny funkce potřebné v projektu.

## 2.3 Organizace v kódu

Z programu se k perifériím přistupuje přes několik singletonů – ImuInterface pro WT901B, BatterInterface pro INA226, GPSInterface pro ublox NEO 7M a ServoControl pro PCA9865. Důvod řešení přes singletony je opět stejný jako u komunikace. Data ze senzorů jsou potřeba v různých částech programu a reálně reprezentují pouze jeden senzor.

Funkce tak nemusí mít celou řadu parametrů a program nepotřebuje jednu centrální třídu, která bude vše organizovat. Mizí tak i problémy s řadou lokálních proměnných a nutností je udržovat aktuální. Jelikož celý program je více vláknový implementují tak singletony ochranu před kolizí několika vláken a vzniku nevalidních dat.

Data ze senzorů, jsou také agregována v třídě telemetry. Ta se stará o nastavení všech senzorů a pak zajišťuje vytvoření packetů pro odeslání klientovi. Ze stejných důvodů jako další třídy i telemetry je implementována jakožto singleton.

O přístup ke kameře se stará třída CameraStreamer. Ta se vytváří dynamicky na základě požadavku od klienta. Jelikož aktuálně není implementována žádná analýza

---

<sup>4</sup>ARIAS, M. *raspberry-pi-ina226*. GitHub. Dostupné z <https://github.com/MarioAriasGa/raspberry-pi-ina226>. [cit 2022-27-2]

<sup>5</sup>ARJAN. *Library PCA9685*. GitHub. Dostupné z <https://github.com/vanvugt/rpidmx512/tree/master/lib-pca9685>. [cit 2022-27-2]

obrazu z kamery přímo na Raspberry Pi proces streamu vznikne jako fork procesu ve kterém je spuštěn stream. Jelikož se jendá o fork při ukončení rodičovského procesu se ukončí i stream a program samotný se o přenos nemusí nijak starat.

## 2.4 Ovládání letadla

O ovládání kontrolních ploch a motoru se stará třída ServoControl. V aktuální podobě přímo zpracovává data z ovladače, která posílá server. Není tak možné jednoduše změnit jaké ovládací prvky budou jak ovládat letadlo.

Na počátku zpracování nových příkazů jsou hodnoty analogové páčky přepočítány do čtvercové plochy. Standardně se totiž mapují ovladače na kruh – páčka čistě nahoru tak má výrazně vyšší hodnotu osy Y, než páčka pod úhlem 45 stupňů, to i přesto, že urazily stejnou vzdálenost. Bez této korekce tak ovládání působilo velmi neresponzivně, kdy praktické byly pouze polohy jih, sever, západ, východ.

V případě konfigurace ocasu do tvaru písmena V je interpretace ovládání relativně jednoduchá. Ocasní plochy (Ruddervator) ovládání bočení a klopení. Plochy na křídlech pak pouze klonění, teoreticky mohou ovládat i klopení, v tomto případě to však nebylo třeba.

### 2.4.1 Autopilot

Projekt implementuje i jednoduchého autopilota, který je schopen držet letovou hladinu. Vyrovnaní řeší dva proporcionalní-integrační-derivační ovladače, které se snaží minimalizovat chybu. Jeden minimalizuje odchylku v klopení, druhý pak odchylku v klonění. Bočení minimalizovat není třeba – pro bočení musí být letadlo nakloněno.

Teoreticky by mohlo jít tento systém rozšířit aby se snažil minimalizovat odchylku nikoliv od původního směru, ale od pomyslného bodu určeného souřadnicemi GPS. Letadlo by tak nejenže letělo více rovně (aktuálně je autopilot náchylní na změnu posunutí v příčné ose), ale mohlo samo doletět na specifikovaný bod.

### 3. Client na ovládání

# Závěr

# Seznam použité literatury

ARIAS, M. *raspberry-pi-ina226*. GitHub. Dostupné z <https://github.com/MarioAriasGa/raspberry-pi-ina226>. [cit 2022-27-2].

ARJAN. *Library PCA9685*. GitHub. Dostupné z [https://github.com/vanvugt/rp\\_idmx512/tree/master/lib-pca9685](https://github.com/vanvugt/rp_idmx512/tree/master/lib-pca9685). [cit 2022-27-2].

TIAN, P. *Arduino-JY901-Serial*. GitHub. Dostupné z <https://github.com/paultian/Arduino-JY901-Serial>. [cit 2022-27-2].

*WT901 Datasheet*. WitMotion Shenzhen Co. Ltd. Dostupné z <https://www.wit-motion.com/gyroscope-module/Witmotion-wt901-ttl-i2c.html>. [cit 2022-27-2].

# Seznam obrázků

2.1	Schéma zapojení obvodu . . . . .	4
2.2	Reálné zapojení . . . . .	5
2.3	Detail Raspberry Pi . . . . .	5
2.4	Detail těla dronu . . . . .	6
2.5	Celý dron . . . . .	6
2.6	Struktura souborů frontendu, vlastní tvorba . . . . .	7