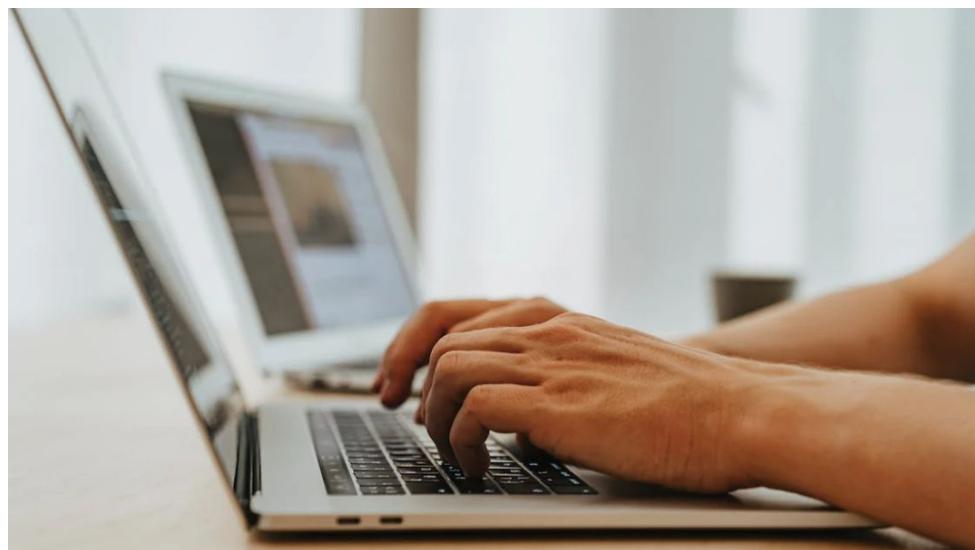




# ZÁVĚREČNÁ STUDIJNÍ PRÁCE

## dokumentace

### Auto na dálkové ovládání



**Autor:** Denis Adamčík  
**Obor:** 18-20-M/01 INFORMAČNÍ TECHNOLOGIE  
se zaměřením na počítačové sítě a programování  
**Třída:** IT4  
**Školní rok:** 2025/26



## **Poděkování**

Rád bych poděkoval panu učiteli Mgr. Marcelu Godovskému za nápady a součástky potřebné k projektu.

## **Prohlášení**

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým a prezentačním účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 1. 1. 2026

.....  
Podpis autora

## **Abstrakt**

Tento ročníkový projekt popisuje návrh a realizaci RC autíčka ovládaného na dálku pomocí Wi-Fi, postaveného na mikrokontrolérech ESP8266. Projekt se skládá ze dvou částí: ovladače, který pomocí tlačítek generuje příkazy, a přijímače v autíčku, který přijímá tyto příkazy a převádí je na signály pro řízení dvou DC motorů přes H-můstky. Komunikace mezi ovladačem a autíčkem je realizována přes Wi-Fi protokol ESP-NOW, což umožňuje bezdrátové ovládání bez potřeby klasické sítě. Cílem projektu bylo demonstrovat základní principy bezdrátové komunikace, řízení pohybu a implementace jednoduchého dálkového ovládání v embedded systému.

## **Klíčová slova**

Esp8266, ESP-NOW, modelování, sestavení, ...

## **Abstract**

This year-end project presents the design and implementation of a Wi-Fi remote-controlled car, based on ESP8266 microcontrollers. The system consists of two main components: a controller, which sends button-driven commands, and a receiver installed on the car, which interprets these commands and converts them into control signals for two DC motors using H-bridge drivers. The communication between the controller and the car is carried out via the ESP-NOW Wi-Fi protocol, enabling wireless operation without a traditional network infrastructure. The aim of the project is to demonstrate fundamental concepts of wireless communication, motion control, and remote operation in an embedded system.

## **Keywords**

Esp8266, ESP-NOW, modeling, assembling ...

# Obsah

<b>Úvod</b>	<b>2</b>
<b>1 Hardware</b>	<b>3</b>
1.1 Použitý mikrokontrolér . . . . .	3
1.2 Návrh schématu zapojení . . . . .	4
1.3 Hardware pro ovladač . . . . .	5
1.4 Hardware autíčka . . . . .	7
<b>2 3D modelování krabičky</b>	<b>9</b>
2.1 3D software . . . . .	9
2.2 3D modely . . . . .	9
<b>3 Software</b>	<b>11</b>
3.1 Bezdrátová komunikace ESP-NOW . . . . .	11
3.2 Ovladač . . . . .	13
3.3 Přijímací jednotka . . . . .	16
<b>Závěr</b>	<b>20</b>

# Úvod

Už od dětství mě fascinovaly věci na dálkové ovládání – auta, letadla, roboti. Bavilo mě sledovat, jak se pohybují na základě neviditelných signálů, a snil jsem o tom, že jednou vytvořím vlastní zařízení, které bude reagovat na mé pokyny. Tento ročníkový projekt mi dal příležitost proměnit dětské nadšení v konkrétní technický výstup.

Pro samotné provedení projektu jsem se rozhodl vytvořit malé autíčko ovládané pomocí mikrokontroléru ESP8266. Díky pomoci pana učitele Mgr. Marcella Godovského, který mi poskytl potřebné elektronické součástky, jsem mohl začít s konstrukcí. Autíčko je navrženo tak, aby reagovalo na signály z ovladače, což umožňuje jeho plynulý pohyb.

Dokumentace obsahuje použité komponenty, zapojení a způsob programování mikrokontroléru. Kromě toho jsou zahrnuty i možné vylepšení projektu. Tato dokumentace slouží jako průvodce pro každého, kdo má zájem o pochopení a reprodukci podobného projektu s využitím ESP technologie.

# 1 HARDWARE

## 1.1 POUŽITÝ MIKROKONTROLÉR

V rámci projektu byl použit stejný typ mikrokontroléru jak v ovladači, tak v samotném autíčku, konkrétně modul ESP8266. Použití shodné platformy v obou částech systému zjednodušuje vývoj softwaru, ladění programu i vzájemnou komunikaci mezi ovladačem a řízeným zařízením.

Mikrokontrolér ESP8266 byl zvolen především díky integrovanému Wi-Fi modulu, který umožňuje bezdrátovou komunikaci mezi ovladačem a autíčkem bez nutnosti použití externích komunikačních modulů. Tato vlastnost snižuje složitost zapojení a zvyšuje spolehlivost celého systému.

Další výhodou mikrokontroléru ESP8266 je jeho dostatečný výpočetní výkon a podpora programování v prostředí Arduino IDE, což umožňuje rychlý vývoj a snadnou úpravu řídicího programu. Díky široké komunitní podpoře je k dispozici velké množství knihoven a ukázkových řešení, která usnadňují implementaci potřebných funkcí.

V ovladači je mikrokontrolér využit zejména pro snímání stavů ovládacích tlačítek a odesílání řídicích signálů. V autíčku pak zajišťuje příjem těchto dat, jejich zpracování a řízení motorů prostřednictvím motorového driveru. Konkrétní zapojení a funkce mikrokontroléru v jednotlivých částech projektu jsou popsány v následujících kapitolách.

Tabulka 1.1: Základní parametry mikrokontroléru ESP8266

Parametr	Hodnota
Napájecí napětí	3,3 V
Komunikační rozhraní	Wi-Fi
Počet GPIO pinů	11
Podpora PWM	Ano
Programovací prostředí	PlatformIO
Použití v projektu	Ovladač i autíčko

## **1.2 NÁVRH SCHÉMATU ZAPOJENÍ**

### **1.2.1 Výběr nástroje**

Pro návrh elektrického schématu a plánování zapojení jednotlivých komponent byl v projektu použit online nástroj EasyEDA. Tento software byl zvolen především díky své dostupnosti, přehlednému uživatelskému rozhraní a jednoduchému ovládání, které umožňuje rychlé vytváření schémat bez nutnosti složité instalace nebo nastavování vývojového prostředí.

### **1.2.2 Výhody použití EasyEDA**

EasyEDA nabízí rozsáhlou knihovnu hotových elektronických součástek, což výrazně urychluje proces návrhu zapojení a snižuje riziko chyb při ručním kreslení schématu. Další výhodou je možnost snadné úpravy zapojení v průběhu vývoje projektu, což je důležité zejména při testování různých variant zapojení a ladění celého systému.

### **1.2.3 Plánování zapojení**

Použití nástroje EasyEDA umožnilo přehledně naplánovat propojení mikrokontroléru s ostatními komponentami ještě před samotnou realizací zapojení. Díky tomu bylo možné odhalit potenciální chyby v návrhu a optimalizovat zapojení, což vedlo k efektivnější montáži a lepší přehlednosti výsledného zapojení.

## **1.3 HARDWARE PRO OVLADAČ**

Původně jsem měl v plánu použít joystick HW-504. Po zjištění, že mikrokontrolér ESP8266 1.1 disponuje pouze jedním analogovým vstupem, bylo však nutné návrh upravit. Joystick jsem proto mohl využít pouze pro řízení rychlosti, zatímco zatáčení jsem musel řešit pomocí tlačítek.

Po dalších technických komplikacích a konzultaci s panem učitelem Mgr. Marcelem Gordovajským jsem se nakonec rozhodl celý ovládací systém zjednodušit a nahradit joystick čtyřmi samostatnými tlačítka. Toto řešení se ukázalo jako spolehlivější, přehlednější a lépe přizpůsobené možnostem použitého mikrokontroléra.

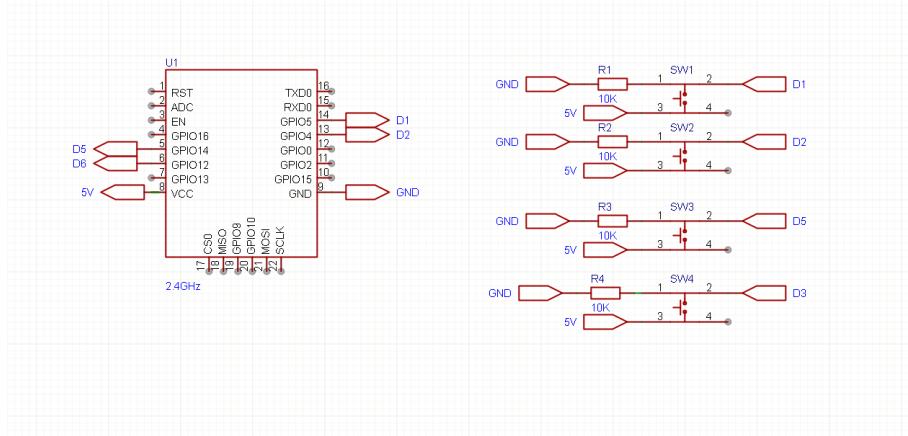
### **1.3.1 Výhody zapojení**

Zapojení ovladače typu sender do projektu přináší výrazné zjednodušení komunikace mezi jednotlivými částmi systému. Umožňuje centralizované odesílání signálů nebo dat, což zvyšuje přehlednost kódu a usnadňuje jeho rozšiřování. Díky jasně definovanému směru toku informací se minimalizuje riziko chyb způsobených nejednoznačnou komunikací mezi moduly. Sender také podporuje lepší modularitu — jednotlivé komponenty mohou být vyvíjeny, testovány a měněny nezávisle, protože se spoléhají na jednotný způsob předávání zpráv. Celkově tak integrace ovladače zvyšuje stabilitu, udržovatelnost a škálovatelnost celého projektu.

### **1.3.2 Nevýhody zapojení**

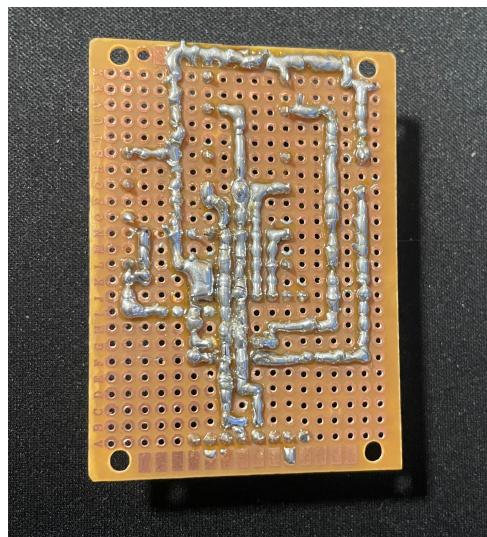
Nevýhodou použitého ovladače (sender) je omezená funkcionálnita řízení, zejména nemožnost plynulé změny rychlosti. Ovládání je realizováno pouze pomocí digitálních tlačítek, která umožňují jen základní povely jako jízda vpřed, vzad nebo zastavení, bez možnosti regulace výkonu motoru. To znamená, že vozidlo pracuje pouze s pevně nastavenou rychlosťí, což snižuje přesnost a komfort ovládání

### 1.3.3 Schema ovladače



Obrázek 1.1: Schema sender

### 1.3.4 Realné zapojení ovladače



Obrázek 1.2: Ručně pájená deska s plošnými spoji

### 1.3.5 Využitý hardware pro ovladač

- 4x mikrospínač
- 1x **ESP8266**
- 4x 10 k $\Omega$  rezistor
- 2x female header (8 pinů)

## **1.4 HARDWARE AUTÍČKA**

Pro konstrukci autíčka byl zvolen jednoduchý a spolehlivý hardware, který umožňuje snadné řízení i případné rozšiřování projektu. Bezdrátový přijímač ES2866 1.1 poskytuje stabilní komunikaci mezi ovladačem a vozidlem, což je klíčové pro přesné reakce při jízdě. H-můstek L9110S byl vybrán díky své kompaktnosti, nízké ceně a schopnosti efektivně řídit dva stejnosměrné motorky nezávisle na sobě. Použité DC motorky jsou lehké, energeticky nenáročné a ideální pro malé mobilní platformy, kde je důležitá jednoduchost zapojení i dostatečný výkon pro pohyb. Všechny tyto části mám propůjčené od pana učitele Mgr. Marcella Godovského.

### **1.4.1 Výhody zapojení**

Použité zapojení přijímače (receiver) přináší přehlednou a logickou strukturu celého systému. Mikrokontrolér ESP8266 přijímá řídicí data z ovladače a na jejich základě přímo ovládá H-můstek, který řídí chod motorů. Toto řešení minimalizuje zpoždění mezi přijetím povelu a reakcí vozidla, což je důležité pro plynulé ovládání. Další výhodou je modularita zapojení, kdy lze jednotlivé části systému snadno upravit nebo rozšířit bez nutnosti zásadních změn v celé konstrukci. Zapojení je také nenáročné na počet součástek, čímž se snižuje riziko poruch a zjednodušuje údržba.

### **1.4.2 Nevýhody zapojení**

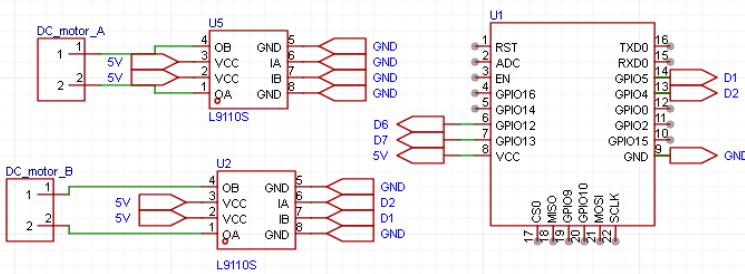
Nevýhodou zvoleného řešení je omezený výkon použitého H-můstku L9110S, který není vhodný pro výkonnější motory nebo vyšší zatížení. To omezuje možnosti dalšího rozšíření projektu, například o těžší konstrukci nebo vyšší rychlosť vozidla. Dalším omezením je absence zpětné vazby z motorů, což znamená, že systém nedokáže sledovat skutečnou rychlosť nebo zatížení pohonu. Řízení vozidla je tedy založeno pouze na přijatých povelích bez další kontroly, což může mít negativní vliv na přesnost pohybu v náročnějších podmírkách.

### 1.4.3 Napajení

Napájení celého systému je řešeno pomocí čtyř AA baterií zapojených do série. Toto zapojení poskytuje napětí přibližně 6 V, které je vhodné pro napájení použitých DC motorů i dalších elektronických součástí. Zvolené řešení bylo vybráno především pro svou jednoduchost, snadnou dostupnost baterií a možnost rychlé výměny bez nutnosti použití externího napájecího zdroje.

Napájecí napětí je dále distribuováno k jednotlivým částem systému, přičemž mikrokontrolér ESP8266 je napájen přes svůj integrovaný stabilizátor, který zajišťuje požadované provozní napětí. Motory jsou napájeny přímo z bateriového zdroje prostřednictvím H-můstku L9110S, což umožňuje efektivní přenos energie a dostatečný výkon pro pohyb vozidla.

### 1.4.4 Schema zapojení



Obrázek 1.3: Schema autička

## **2 3D MODELOVÁNÍ KRABIČKY**

### **2.1 3D SOFTWARE**

Při výběru modelovacího softwaru pro můj projekt jsem měl na výběr mezi několika možnostmi, mezi nimiž byly klíčovými kandidáty Onshape, Sketchfab a Autodesk Inventor. Po důkladném zvážení všech faktorů jsem se rozhodl pro Autodesk Inventor.

#### **2.1.1 Výhody Autodesk Inventoru**

Rozhodl jsem se využít Autodesk Inventor díky mé předchozí zkušenosti s tímto programem. Tato volba mi umožnila efektivní modelování bez nutnosti učení se nového softwaru. Díky studentské licenci poskytnuté školou jsem nemusel platit za software. Autodesk Inventor, specializovaný na inženýrství a konstrukci, odpovídá potřebám mého projektu díky parametrickému modelování a vhodným nástrojům. I když může být v uměleckém modelování omezený ve srovnaní s některými ostatními softvary, pro mé konkrétní potřeby a s ohledem na rychlý a efektivní postup modelování s minimální časovou investicí byla tato volba vhodná a optimální.

### **2.2 3D MODELY**

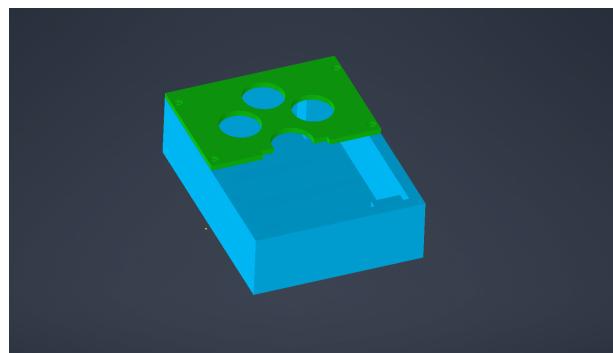
Pro konstrukci krabičky bylo nutné vymodelovat dva samostatné 3D modely – spodní část a horní víko. Spodní díl slouží jako nosná část krabičky a je navržen tak, aby poskytoval dostatečný prostor pro umístění elektronických komponent a jejich upevnění. Horní díl krabičky je navržen jako odnímatelné víko, které obsahuje otvory pro ovládací prvky a díry pro přichycení. Rozdělení krabičky na dva samostatné modely usnadňuje jak samotnou montáž elektroniky, tak i výrobu pomocí 3D tisku a případnou údržbu zařízení.

Při modelování krabičky byly využity základní nástroje programu Autodesk Inventor, zejména náčrt (Sketch), vysunutí (Extrude), odebrání materiálu (Cut) a zaoblení hran (Fillet). Model byl vytvářen parametricky, což umožňuje snadnou úpravu rozměrů krabičky v případě změny velikosti nebo typu použitých elektronických komponent.

Rozměry krabičky a rozmístění otvorů byly navrženy s ohledem na konkrétní elektronické součástky použité v projektu. Ovládací prvky jsou umístěny v horním víku tak, aby byly snadno přístupné při běžném používání ovladače, zatímco vnitřní prostor spodního dílu zajišťuje bezpečné uložení desky s mikrokontrolérem a propojení vodičů.

Horní a spodní díl krabičky jsou spojeny pomocí šroubů, které zajišťují pevné uchycení a zároveň umožňují snadné rozebrání krabičky v případě potřeby údržby, opravy nebo budoucích úprav zapojení.

Model krabičky byl navržen s ohledem na následnou výrobu pomocí 3D tisku. Byly zohledněny minimální tloušťky stěn, orientace tisku a omezení tiskového procesu, aby byla zajištěna dostatečná pevnost a funkčnost výsledného výrobku. Výsledný 3D model krabičky je znázorněn na obrázku 2.1.



Obrázek 2.1: Model krabičky

## 3 SOFTWARE

### 3.1 BEZDRÁTOVÁ KOMUNIKACE ESP-NOW

ESP-NOW [1] je bezdrátový komunikační protokol vyvinutý společností Espressif Systems pro mikrokontroléry řady ESP32 a ESP8266. Tento protokol umožňuje výměnu dat mezi zařízeními bez použití klasické Wi-Fi sítě nebo přístupového bodu, čímž se dosahuje nízké latence a malé režie komunikace. Využívá přímou peer-to-peer komunikaci pomocí MAC adres jednotlivých zařízení a je plně podporován v prostředí Arduino IDE.

#### 3.1.1 Princip funkce

ESP-NOW funguje jako bezspojoval (connectionless) protokol, kde zařízení komunikují přímo mezi sebou bez navazování tradičního Wi-Fi spojení. Data jsou odesílána ve formě krátkých paketů s maximální délkou až 250 bajtů. Protokol podporuje různé režimy komunikace:

- komunikace jeden-na-jeden,
- jeden-na-více (broadcast),
- více-na-jeden.

ESP-NOW může pracovat současně s Wi-Fi, pokud je zařízení ve správném režimu, což umožňuje kombinovat rychlou bezdrátovou komunikaci s přístupem do sítě. :contentReference[oaicite:1]index

#### 3.1.2 Výhody

Mezi hlavní výhody ESP-NOW patří:

- velmi nízká latence a režie při přenosu dat,
- nízká spotřeba energie ve srovnání s plnohodnotnou Wi-Fi sítí,
- možnost šifrování přenášených paketů pro větší bezpečnost,

ESP-NOW je ideální pro projekty vyžadující rychlou výměnu malých datových bloků mezi více zařízeními bez nutnosti centrálního routeru. :contentReference[oaicite:2]index=2

### **3.1.3 Omezení**

Protokol má také některá omezení, například:

- omezená velikost přenášených dat (maximálně 250 bajtů),
- maximální počet zařazených peerů při zabezpečené komunikaci,
- kratší dosah ve srovnání s klasickou Wi-Fi sítí při stejném výkonu.

Tyto limity je třeba zohlednit při návrhu systémů, které by vyžadovaly přenos větších objemů dat nebo větší topologickou flexibilitu. :contentReference[oaicite:3]index=3

### **3.1.4 Implementace v Arduino IDE**

Pro využití ESP-NOW v prostředí Arduino IDE je potřeba nejprve nainstalovat podporu pro desky ESP8266. Poté lze pomocí knihoven `esp_now.h` a `WiFi.h` 3.2.1 nakonfigurovat zařízení jako vysílač, přijímač či obojí. Komunikace probíhá prostřednictvím callback funkcí, které zpracovávají odesílání a příjem dat. Každé zařízení musí být předem spárováno se svými komunikačními protějšky pomocí jejich MAC adresy.

## 3.2 OVLADAČ

Tato část dokumentace popisuje software pro vysílací jednotku založenou na mikrokontroleru ESP8266. Program je napsán v jazyce C++ a je určen pro vývojové prostředí Arduino IDE. Software zajišťuje čtení stavu čtyř tlačítek a jejich bezdrátový přenos pomocí protokolu ESP-NOW.

### 3.2.1 Použité knihovny

Na začátku programu jsou zahrnuty potřebné knihovny pro práci s Wi-Fi a ESP-NOW komunikací.

```
1 #include <ESP8266WiFi.h>
2 #include <espnow.h>
```

Kód 3.1: Zahrnutí knihoven

### 3.2.2 Definice pinů tlačítek

Tlačítka jsou připojena k GPIO pinům mikrokontroleru. V kódu jsou použity interní pull-up rezistory, proto je logická úroveň LOW považována za stisknuté tlačítko.

```
1 #define BUTTON1_PIN 14 // D5
2 #define BUTTON2_PIN 12 // D6
3 #define BUTTON3_PIN 13 // D7
4 #define BUTTON4_PIN 4 // D8
```

Kód 3.2: Definice GPIO pinů tlačítek

### 3.2.3 Datová struktura

Pro přenos informací o stavu tlačítek je použita struktura, která obsahuje čtyři logické proměnné. Každá proměnná reprezentuje jedno tlačítko.

```
1 typedef struct struct_message {  
2     bool button1;  
3     bool button2;  
4     bool button3;  
5     bool button4;  
6 } struct_message;
```

Kód 3.3: Struktura odesílaných dat

### 3.2.4 Inicializace programu

Ve funkci `setup()` je inicializována sériová komunikace, nastaven Wi-Fi režim, GPIO piny a ESP-NOW komunikace. Zařízení je nastaveno do role řídicí jednotky (controller).

```
1 void setup() {  
2     Serial.begin(115200);  
3     WiFi.mode(WIFI_STA);  
4  
5     pinMode(BUTTON1_PIN, INPUT_PULLUP);  
6     pinMode(BUTTON2_PIN, INPUT_PULLUP);  
7     pinMode(BUTTON3_PIN, INPUT_PULLUP);  
8     pinMode(BUTTON4_PIN, INPUT_PULLUP);  
9  
10    if (esp_now_init() != 0) {  
11        Serial.println("Chyba inicializace ESP-NOW");  
12        return;  
13    }  
14  
15    esp_now_set_self_role(ESP_NOW_ROLE_CONTROLLER);  
16    esp_now_register_send_cb(OnDataSent);  
17    esp_now_add_peer(broadcastAddress, ESP_NOW_ROLE_SLAVE, 1, NULL, 0);  
18 }
```

Kód 3.4: Inicializace ESP-NOW a pinů

### 3.2.5 Hlavní smyčka programu

V nekonečné smyčce `loop()` dochází ke čtení stavu jednotlivých tlačítek, uložení hodnot do struktury a jejich odeslání přijímací jednotce. Pro ladění je stav tlačítek vypisován do sériového monitoru.

```
1 void loop() {  
2     myData.button1 = !digitalRead(BUTTON1_PIN);  
3     myData.button2 = !digitalRead(BUTTON2_PIN);  
4     myData.button3 = !digitalRead(BUTTON3_PIN);  
5     myData.button4 = !digitalRead(BUTTON4_PIN);  
6  
7     esp_now_send(broadcastAddress, (uint8_t *)&myData, sizeof(myData));  
8  
9     Serial.print("B1: "); Serial.print(myData.button1);  
10    Serial.print(" | B2: "); Serial.print(myData.button2);  
11    Serial.print(" | B3: "); Serial.print(myData.button3);  
12    Serial.print(" | B4: "); Serial.println(myData.button4);  
13  
14    delay(100);  
15 }
```

Kód 3.5: Čtení tlačítek a odesílání dat

### 3.3 PŘIJÍMACÍ JEDNOTKA

Tato část dokumentace popisuje software přijímací jednotky založené na mikrokontroleru ESP8266, která ovládá pohyb vozidla pomocí dvou stejnosměrných motorů. Program je napsán v jazyce C++ a určen pro vývojové prostředí Arduino IDE. Komunikace mezi vysílací a přijímací jednotkou je realizována bezdrátově pomocí protokolu ESP-NOW.

#### 3.3.1 Použité knihovny

Na začátku programu jsou zahrnutы knihovny nezbytné pro práci s Wi-Fi rozhraním a ESP-NOW komunikací.

```
1 #include <ESP8266WiFi.h>
2 #include <espnow.h>
```

Kód 3.6: Zahrnutí knihoven

#### 3.3.2 Datová struktura

Pro příjem informací o stavu tlačítek je použita struktura shodná se strukturou vysílací jednotky. Jednotlivé logické proměnné reprezentují směrová tlačítka ovladače.

```
1 typedef struct struct_message {
2     bool button1; // vlevo
3     bool button2; // vpřed
4     bool button3; // vpravo
5     bool button4; // vzad
6 } struct_message;
7
8 struct_message myData;
```

Kód 3.7: Struktura přijímaných dat

### 3.3.3 Definice pinů motorů

Vozidlo je poháněno dvěma stejnosměrnými motory (pravý a levý), které jsou řízeny pomocí PWM signálů. Každý motor má dva řídicí piny pro změnu směru otáčení.

```
1 // Pravý motor  
2 #define A1_A 14 // vpřed  
3 #define A1_B 12 // vzad  
4  
5 // Levý motor  
6 #define B1_A 4 // vzad  
7 #define B1_B 5 // vpřed
```

Kód 3.8: Definice GPIO pinů motorů

### 3.3.4 Řízení rychlosti motorů

Rychlosť motorů je řízena pomocí PWM signálu. Pro přímý pohyb je použita vyšší hodnota rychlosti, zatímco při zatáčení je jeden z motorů zpomalen.

```
1 const int speed = 600;  
2 const int turnSpeed = 200;
```

Kód 3.9: Nastavení rychlostí

Pro zjednodušení řízení motorů je vytvořena pomocná funkce, která nastavuje PWM hodnoty jednotlivých pinů.

```
1 void setMotorsPWM(int a1a, int a1b, int b1a, int b1b) {  
2     analogWrite(A1_A, a1a);  
3     analogWrite(A1_B, a1b);  
4     analogWrite(B1_A, b1a);  
5     analogWrite(B1_B, b1b);  
6 }
```

Kód 3.10: Funkce pro ovládání motorů

### 3.3.5 Příjem dat pomocí ESP-NOW

Příjem dat z vysílací jednotky je zajištěn pomocí callback funkce, která je automaticky volána při příjetí zprávy. Data jsou zkopirována do struktury a následně vyhodnocena.

```
1 void OnDataRecv(uint8_t *mac, uint8_t *incomingData, uint8_t len) {  
2     memcpy(&myData, incomingData, sizeof(myData));  
3     lastRecvTime = millis();  
4     dataReceived = true;  
5     ...  
6 }
```

Kód 3.11: Callback funkce pro příjem dat

### 3.3.6 Logika ovládání vozidla

Na základě kombinací stisknutých tlačítek je určován směr a způsob pohybu vozidla. Program podporuje:

- přímý pohyb vpřed a vzad,
- zatáčení na místě,
- plynulé zatáčení v oblouku,
- bezpečnostní zastavení při stisku tří a více tlačítek.

Pokud nejsou přijímána žádná data nebo dojde ke ztrátě signálu, vozidlo se automaticky zastaví.

### 3.3.7 Inicializace programu

Ve funkci `setup()` je inicializována sériová komunikace, nastaven Wi-Fi režim, nakonfigurovány výstupní piny motorů a spuštěna ESP-NOW komunikace v režimu přijímací jednotky (slave).

```
1 void setup() {  
2     Serial.begin(115200);  
3     WiFi.mode(WIFI_STA);  
4  
5     pinMode(A1_A, OUTPUT);  
6     pinMode(A1_B, OUTPUT);  
7     pinMode(B1_A, OUTPUT);  
8     pinMode(B1_B, OUTPUT);  
9  
10    setMotorsPWM(0, 0, 0, 0);  
11  
12    esp_now_init();  
13    esp_now_set_self_role(ESP_NOW_ROLE_SLAVE);  
14    esp_now_register_recv_cb(OnDataRecv);  
15}
```

Kód 3.12: Inicializace přijímací jednotky

### 3.3.8 Hlavní smyčka programu

V hlavní smyčce `loop()` je kontrolováno, zda jsou přijímána data. Pokud dojde k překročení časového limitu, motory jsou automaticky zastaveny, což zvyšuje bezpečnost provozu vozidla.

```
1 void loop() {  
2     if (!dataReceived || millis() - lastRecvTime > timeout) {  
3         setMotorsPWM(0, 0, 0, 0);  
4     }  
5 }
```

Kód 3.13: Kontrola ztráty signálu

# ZÁVĚR

Cílem této závěrečné studijní práce bylo navrhnout a realizovat funkční model autíčka na dálkové ovládání s využitím mikrokontrolérů ESP8266 a bezdrátové komunikace pomocí protokolu ESP-NOW. Tento cíl byl úspěšně splněn – vznikl kompletní systém skládající se z ovladače a přijímací jednotky, které spolu spolehlivě komunikují a umožňují ovládání pohybu vozidla v reálném čase.

V průběhu práce jsem se seznámil s návrhem hardwaru i softwaru embedded systémů. Naučil jsem se pracovat s mikrokontrolérem ESP8266, navrhovat schémata zapojení a implementaci bezdrátové komunikace bez použití klasické Wi-Fi sítě. Zajímavou zkušeností bylo také řešení omezení hardwaru, například malého počtu analogových vstupů, což vedlo k úpravě původního návrhu ovládání.

Závěrem lze říci, že projekt splnil stanovené cíle a poskytl mi cenné praktické zkušenosti v oblasti elektroniky, programování a bezdrátové komunikace. Získané znalosti považuji za dobrý základ pro další studium i budoucí projekty v oblasti informačních technologií a embedded systémů.

## LITERATURA

- [1] „Getting Started with ESP-NOW (ESP8266 with Arduino IDE)“, Random Nerd Tutorials, dostupné z: <https://randomnerdtutorials.com/esp-now-esp32-arduino-ide/>.
- [2] „ESP8266 Pinout Reference: Which GPIO Pins Should You Use?“, Random Nerd Tutorials, dostupné z: <https://randomnerdtutorials.com/esp8266-pinout-reference-gpios/>

## Seznam obrázků

1.1	Schema sender . . . . .	6
1.2	Ručně pájená deska s plošnými spoji . . . . .	6
1.3	Schema autička . . . . .	8
2.1	Model krabičky . . . . .	10

## Seznam tabulek

1.1	Základní parametry mikrokontroléru ESP8266 . . . . .	3
-----	--	---