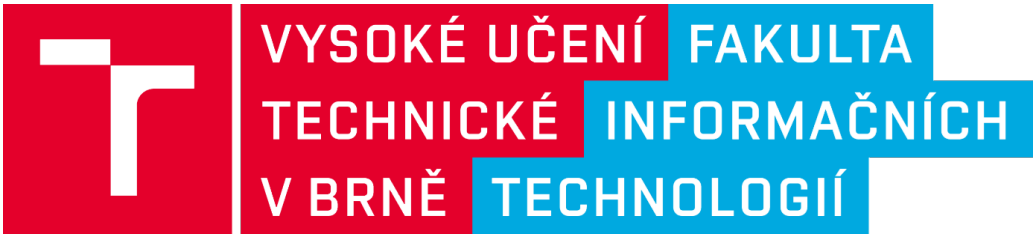


Mikroprocesorové a vestavěné systémy

ESP32: Řízení otáček BLDC motoru s měřením otáček

30. listopadu 2022

Tomáš Souček, xsouce15

Obsah

1 Úvod

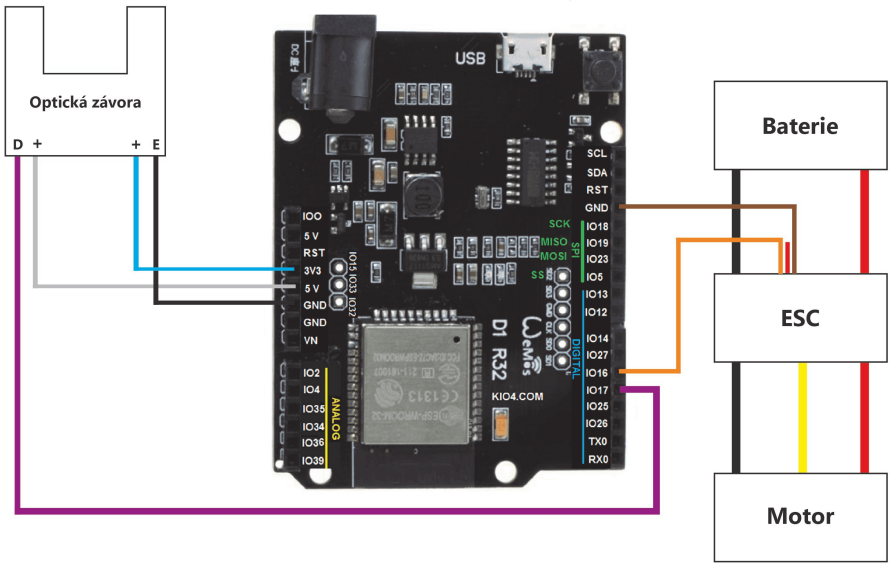
Cílem tohoto projektu bylo vytvořit program pro řízení otáček bezkartáčového BLDC motoru za pomoci optiké závory. K řešení projektu byla vybrána platforma ESP32.

2 Návrh a implementace

Pro implementaci projektu byl vybrán následující HW:

- BLDC motor 2212-13T
- ESC regulátor otáček pro BLDC 30A
- 3článková baterie (3S1P) DroneArt akupack Li-Pol 11.1 V, 2300 mAh, 45 C Softcase XT60H
- Optická závora TCST1103
- Propojovací dráty

Jednotlivé komponenty byly, na základě poskytnutého schéma k ESP32 [2], zapojeny následovně:



Obrázek 1: Schéma zapojení

Na základě konzultace s vedoucím projektu bylo řešení realizováno na rozhraní Arduino. Pro správné fungování aplikace byla potřeba naimportovat knihovnu `ESP32Servo`. Implementační kód se nachází v souboru `main.ino`.
Hlavní funkcí programu je zvyšování otáček motoru, pokud je do optické závory vložen předmět přerušující paprsek diody, po odstranění předmětu se otáčky snižují na minimum.

Ze začátku programu jsou nadefinovány konstanty s čísly pinů, ke kterým jsou připojeny periferie. Odlišností je zde LED dioda na pinu 2, jelikož je zabudovaná, nejedná se o perifeii. V programu slouží jako indikátor stavu optické závory. Následně dochází také k inicializaci hodnot potřebných pro běh programu jako je např. minimální počet otáček motoru, nebo počitadlo průběhů programu.

2.1 setup()

Jedná se o funkci, která je spuštěná v programu těsně před hlavní smyčkou implementovanou ve funkci `loop()`. Slouží k inicializaci módů pinů, jejich propojení s dalšími objekty (`esc.attach(ESC_PIN)`) a inicializaci rychlosti komunikace mezi deskou a výpisovou periférii (v mém případě konzoli v `arduino IDE`). Funkce proběhne pouze 1x za celou dobu běhu programu.

2.2 loop()

Funkce `loop()` vyjadřuje hlavní smyčku programu, která se vykonává až do jeho ukončení. Funkce na začátku běhu programu zkalibruje motor a následně volá funkce na zvyšování či snižování počtu otáček motoru. Funkce `delay()` slouží pouze k tomu, aby byla změna otáček vidět pozvolněji. Na konci smyčky dojde k inkrementaci počitadla běhů smyčky a k nastavení nové pulsní šířky motoru.

2.3 calibrateMotor()

Funkce sloužící pro prvotní kalibraci motoru. Hodnoty byly převzány z referenčního manuálu pro konkrétní, výše zmíněný, ESC regulátor otáček [1].

2.4 increaseSpeed()

Jak již anglický název funkce napovídá, jedná se o funkci, který řídí rychlost otáček motoru. Konkrétně ji zvyšuje. Funkce je volána ve chvíli, kdy je přerušen paprsek v optické závoře, tedy je do ní vložen nějaký předmět. Jako indikaci přerušení paprsku se používá rozsvícení LED diody na pinu 2. Následně vypíše pomocí knihovný funkce `Serial.println(speed)` aktuální pulsní šířku signálů v mikrosekundách. Daná hodnota reprezentuje rychlost motoru. Na konci dojde k ověření, zda nebylo dosáhnuto maximální pulsní šířky. V případě kladné odpovědi dojde ke zvýšení o 25 mikrosekund. Hodnota 25 byla vybrána, aby byla demonstrace zřetelnější.

2.5 decreaseSpeed()

Další funkce, jejíz název je také samovysvětlující, slouží ke snižování otáček motoru. Je volána z hlavní smyčky pokaždé, když se v optické závoře nenachází žádný předmět, který by rušil její vysílaný signál. Funguje obdobnou logikou jako funkce pro zvyšování rychlosti a to tak, že zajistí vyypnutí signalizační LED diody, vypíše současnou pulsní šířku a pokud je větší, než minimální dovolená, pak ubere o 25 mikrosekund. Za minimální pulsní šířku byla v programu vybrána hodnota 1200. Teoreticky by mohla být i nižší, ale pro názornost ukázky byla vybrána hodnota dostatečná pro minimální roztočením motoru. Minimální hodnota však nesmí klesnout pod 1050 mikrosekund, jelikož by se pak motor přepnul opět do kalibračního režimu.

3 Závěr

Cílem projektu bylo seznámit se s programováním mikroprocesorů a vytvářením vestavných systémů. K tomuto účelu byl vytvořen funkční program pro řízení otáček bezkartáčového BLDC motoru za pomoci optiké závory. Testování programu neukázalo implementační nedostatky. Jelikož k předvedení funkčnosti nestačí obrázek, bude funkčnost demonstrována na osobně.

3.1 Autoevaluace

- E — Vybavení na projekt bylo zapůjčeno včasně, do práce jsem se pustil cca měsíc před deadlinem pro odevzdání, projekt splňuje požadavky konzultované s vedoucím. Nevytvářím však nadbytečná rozšíření, tedy **0.8 b**.
- F – Funkčnost a koncepce byla odsouhlasena vedoucím během konzulatce. Projekt jsem řešil s použitím rozhraní Arduino, tudíž **4.5 b**.
- Q – Logika programu je vcelku jednoduchá, tedy bych řekl, že i ovládání je lehce pochopitelné. Kód je čitelný a přehledný s vhodnou mírou dekompozice. **3 b**.
- P – Ukázka jasně demonstruje funkčnost řešení, takže **1 b**.
- D – Dokumentace popisuje zapojení využitých periferií, logiku programu a splňuje i zbylé náležitosti. Jediné, co bych si vytknul, je její délka. Z toho plyne hodnocení **3.5 b**

Výpočet celkového hodnocení je tedy následující:
$\sum = (0.25 + 0.75 \cdot 4.5/5) \cdot (0.8 + 4.5 + 3 + 1 + 3.5) = \mathbf{11.84\ b}$

Reference

[1] Eses czech importer: *ESC regulátor otáček PDF*. [online], [viděno 17.10.2022]. URL <https://dratek.cz/docs/produkty/1/1056/1498208081.pdf>

[2] Ing. Vojtěch Mrázek, Ph.D.: *Projekt - ESP32*. [online], [viděno 17.10.2022]. URL <https://moodle.vut.cz/mod/page/view.php?id=254926>