Robot mobilny klasy Line Follower Mjølner

Robert Kiczke
Kamil Kiełbasa

Koło Naukowe Robotyków KoNaR

Spis treści

1 Wstęp

Mjølner to robot klasy Line-Follower, który został stworzy na warsztatach robotycznych kierowanych przez koło naukowe KoNaR przy Politechnice Wrocławskiej. Naszym celem było wzięcie udziału w zawodach Robotic Arena 2016 ale również możliwość rekrutacji do koła naukowego KoNaR. Naszym opiekunem był Bartek Kurosz. Wkład Bartka w pomoc nad budową robota jest bezcenny, także Jego zdolności tłumaczenia trudnych pojęć w sposób bardzo klarowny jak i cierpliwość. Naszym celem było zbudowanie robota który posiadałby szerokie pole widzenia oraz maksymalnie zoptymalizowaną wagę przy minimalnych wymiarach.

2 Mjølner

2.1 Mechanika

Podstawą kontrukcji robota jest wydzielenie z płytki PCB dwóch oddzielnych modułów. W głowny module znajduje się najważniejsza elektronika, natomiast drugim tylko czujniki odbiciowe. Całość jest połączona listwą węglową. Taka konstrukcja ma wiele zalet, jedną z ważniejszych jest jej prostota. Odległość między obiema płytkami można modyfikować. Nie jest tu potrzebna żadna specjalna kontrukcja dla robota, przez co jest bardzo lekki. W części płytki gdzie znajduje się główna elektronika, już podczas projektowania wydzieliliśmy specjalną przestrzeń dla łożysk. Dzięki temu ciężko było uszkodzić elektronikę podczas montażu.

2.1.1 Koła

Zastosowaliśmy koła poliuretanowe wykonane przez Michała Burdkę. Koła sprawdziły bardzo dobrze, otrzymaliśmy maksymalnie możliwą przyczepność.

2.1.2 Silniki

Na module głównym zamontowane są silniki Polulu 30:1 wraz z enkoderami. Mieliśmy do wyboru także 10:1 lecz postawiliśmy na większy moment siły. Robot było bardzo zwrotny na zakrętach kosztem prędkości oraz przyśpieszenia.

2.1.3 Podsumowanie

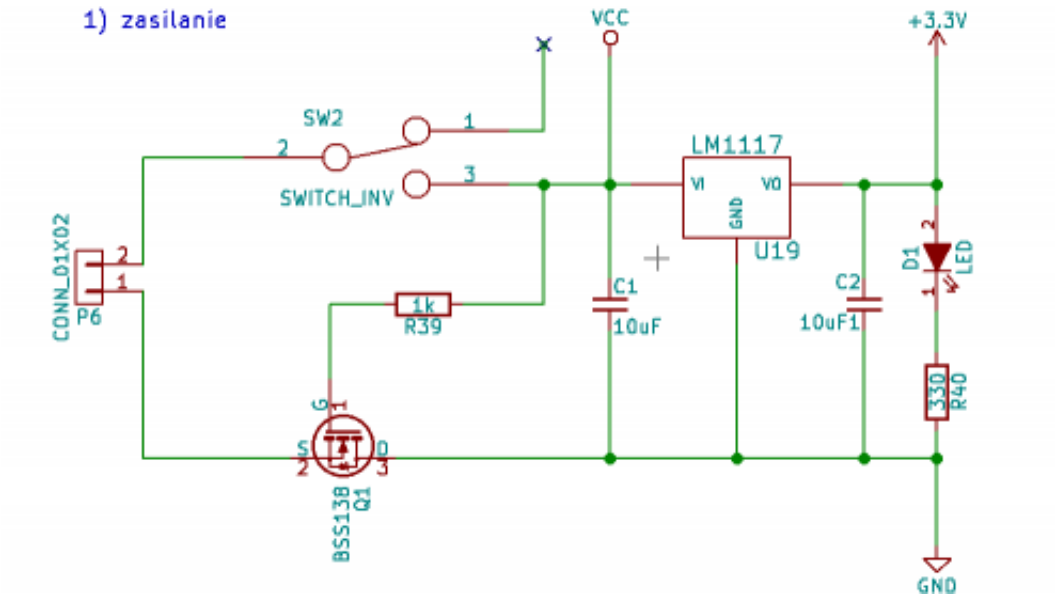
Planowane jest przetestowanie silników z przekładnią 10:1 oraz zamontowanie w bardziej stabliny sposób akumulatora.

2.2 Elektronika

PCB zostało stworzone w programie KiCad. Jest to oprogramowanie open-source. Płytki są dwustronne i tworzone za pomocą fototransferu. Zakładaliśmy minimalizacje płytek, więc elementu są bardzo blisko siebie natomiast unikaliśmy umiejscowania ścieżek zbyt blisko siebie lub w pobliżu krawędzi. Używaliśmy tylko i wyłącznie elementów montowanych powierzchniowo SMD (Surface Mounted Devices).

2.2.1 Zasilanie

Robot jest zasilany za pomocą pakietu Li-Pol, składającym się z dwóch ogniw o sumarycznym napięciu 7.4V. Wszystkie podzespoły pracuję pod napięciem 3.3V natomiast wyjątkiem są silniki, które są zasilane bezpośrednio z akumulatora. Wybralo akumulator Li-Pol ponieważ cechuje się dużą pojemnością jaki i wydajnością prądową przy bardzo małych rozmiarach jak i masie. Do uzyskania pożadanego napięcia 3.3V użyto stabilizatora LM1117DT. Zastosowaliśmy podwójne zabezpieczenie przed odwrotną polaryzacją w postaci złącza z kluczem jak i tranzystora typu N. Do włączania zasilania użyto przełącznika suwakowego. Poprawne zasilanie sygnalizował zielony LED.



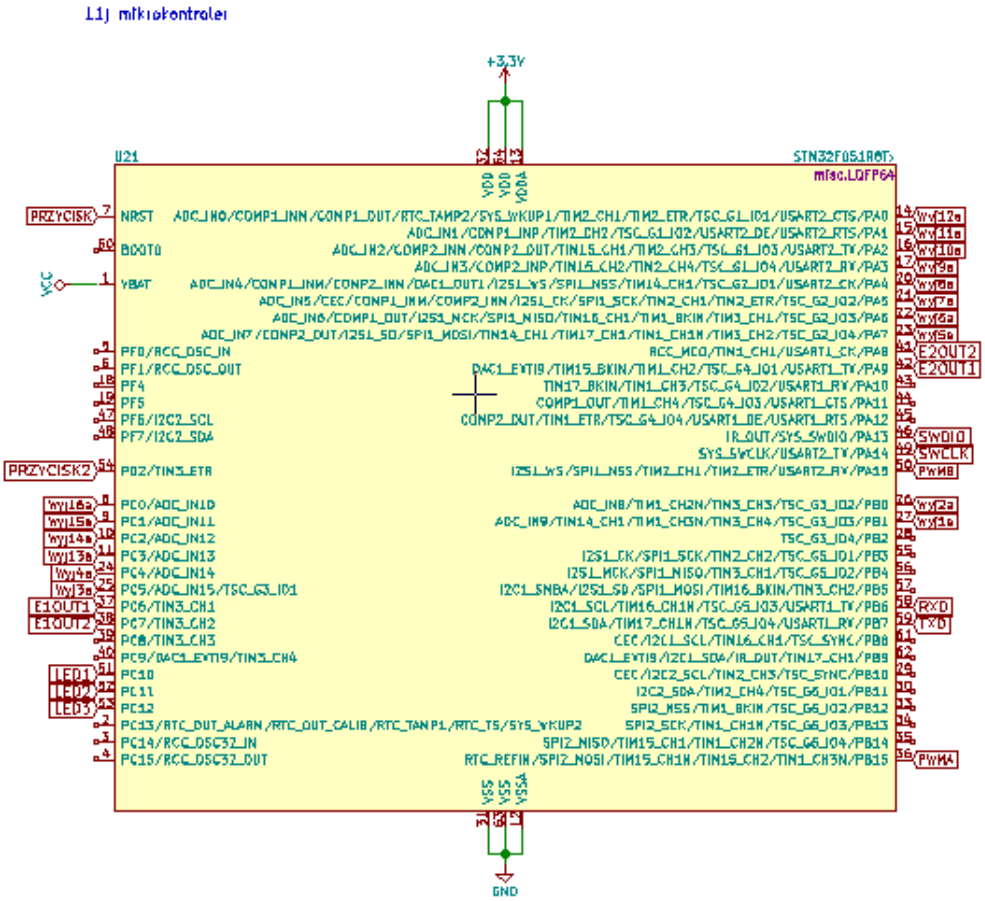
Zasilanie

2.2.2 Mikrokontroler

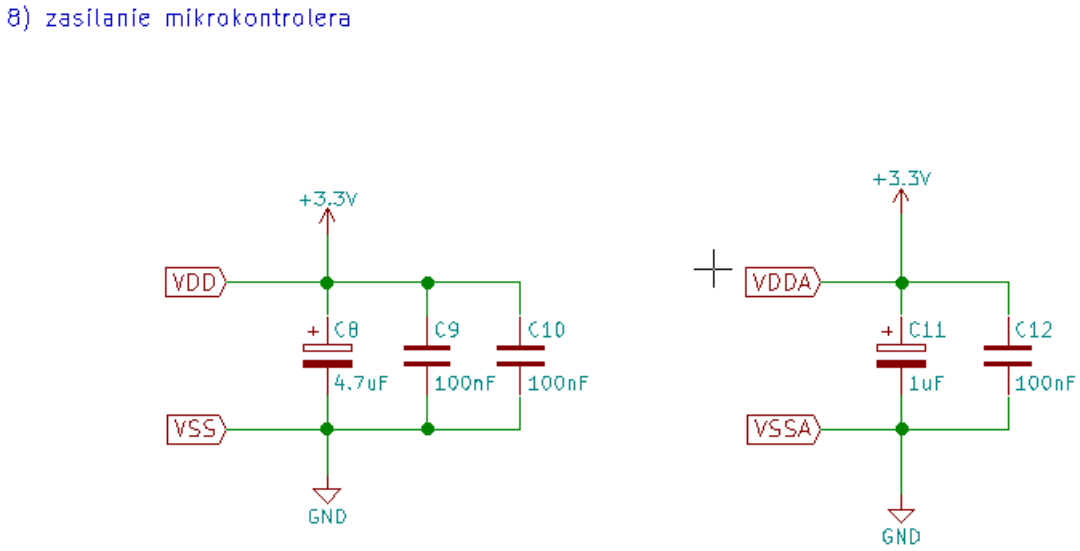
Wybrano mikrokontroler STM32F051R8T6
Specyfikacja:

- Typ układu scalonego: Mikrokontroler ARM
- Pojemność pamięci Flash: 64kB
- Częstotliwość taktowania: 48MHz
- Montaż: SMD
- Liczba wejść/wyjść: 55
- Pojemność pamięci SRAM: 8kB
- Obudowa: LQFP64
- Napięcie zasilania: 2V - 3.6V DC
- Rodzaj architektury: Cortex M0
- Liczba timerów 16-bitowych: 7
- Liczba timerów 32-bitowych: 1
- Interfejs: I2C x2, SPI, UART x2
- Masa: 2.12g

Początkowo mieliśmy użyć mikrokontrolera STM32F051C8T6, różni się tylko ilością dostępnych wejść i wyjść (tj. 39). Trafne było użycie powyższego mikrokontrolera ze względu na wygodniejszą możliwość konfiguracji peryferii.



Mikrokontroler

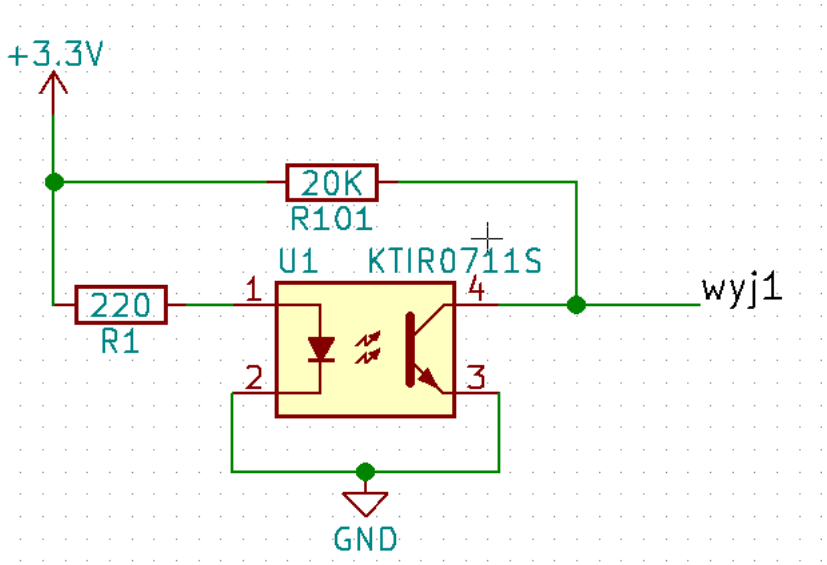


Zasilanie mikrokontrolera

2.2.3 Czujniki

Wybrano czujniki odbiciowe KTIR0711S
Specyfikajca:

- Maksymalne napięcie diody IR: 5 V
- Maksymalny prąd diody IR: 50 mA
- Maksymalne napięcie kolektor-emiter: 30 V
- Maksymalny prąd kolektora: 20 mA
- Obudowa lutowana powierzchniowo (SMD)

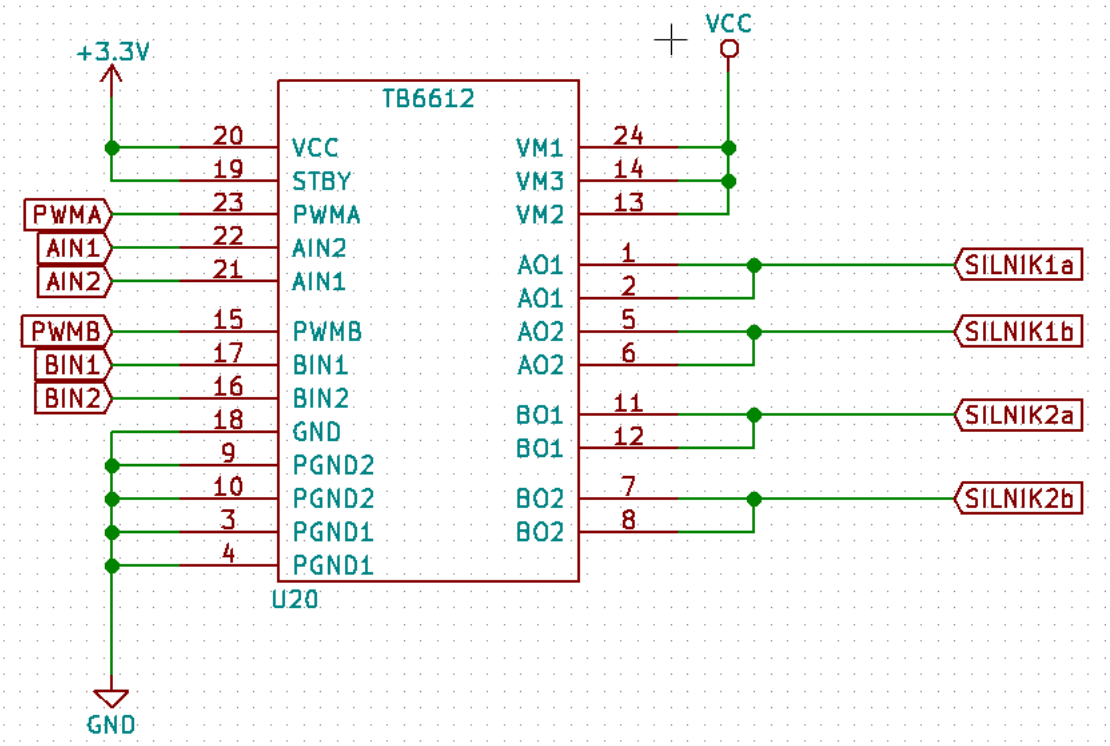


Czujnik odbiciowy

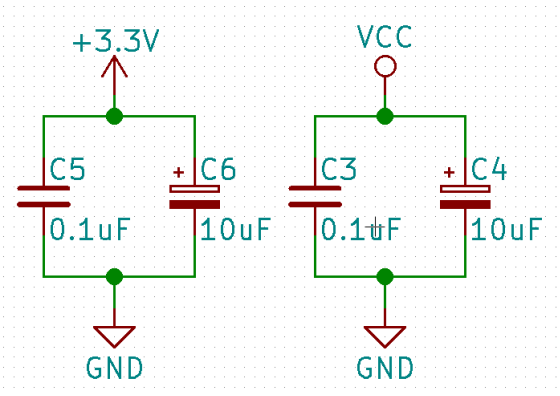
2.2.4 Sterowanie silnikami

Wybrano dwukanałowy sterownik silników TB6612FNG
Specyfikacja:

- Maksymalne napięcie zasilania silników: 4,5 - 15 V
- Prąd ciągły 1,2 A
- Prąd chwilowy: 3,2 A
- Tryb niskiego poboru prądu (Standby)
- Możliwość zmiany kierunku obrotów silnika
- Możliwość szybkiego zatrzymania silnika
- Obudowa: SMD SSOP24
- Kanały sterownika można połączyć aby otrzymać większą wydajność prądową



Mostek H



Zasilanie Mostka H

2.2.5 Interfejs komunikacyjny

Z robotem komunikowaliśmy się za pomocą programatora z płytki Nucleo lub Discovery. Robot posiada trzy diody, z czego wykorzystaliśmy tylko dwie. Pierwsza dioda świecąc wskazywała na płynący prąd w obwodzie. Druga świeciła się po poprawnym wgraniu kodu (ostatnia linija w kodzie zaświecała diodą). Natomiast robot posiada moduł bluetooth lecz przez brak czasu zrezygnowaliśmy z programowania tego modułu zajmując się kalibracją robota.

2.3 Program

Program był pisany w języku C, z wykorzystaniem biblioteki HAL. Korzystano z darmowego środowiska programistycznego dla mikrokontrolerów STM32. Do wygnerowania konfiguracji peryferiów użyto darmowego programu STM32CubeMX. System Workbench for STM32 to darmowe środowisko programistyczne w opraciu o Elipse. Pozwala to na programowanie jak i również debugowanie danych urządzeń. Bezcenne okazało się STMStudio do wizualizowania pracy czujników odbiciowych oraz STM32 ST-LINK Utility jako alternatywa do programowania i łączenia się z mikrokontrolerem.

2.3.1 Konfiguracja peryferii

Do wykrycia czarnej linii użyto ADC korzystające z DMA (Direct Memory Access, DMA (z ang. bezpośredni dostęp do pamięci)). Do regulacji obrotów zostało oparne na generacji sygnału PWM przez timer o częstotliwości taktowania 1000Hz.
Konifguracja timerów
Dane:

- Prescaler (16-bitowy) = 47
- Counter Mode = up
- Counter Period = 999
- Internal Clock Division = no division

Pętla algorytmu sterownia robotem jest wykonywana w przerwaniu timera z częstotliwościa 100Hz.

2.3.2 Algorytm sterowania

Do sterowania zastosowaliśmy algorytm PD (proporcjonalo-różniczkujący). Nie stosowaliśmy członu całkującego ponieważ komplikuje on układ i nie

wnosi poprawy w sterowaniu. Regulator PD jest wykonywany w przerwaniu timera z częstotliwością 100Hz. Człon proporcjonalny jest odpowiedzialny za natychmiastową reakcję przy zakłóceniu jakim jest zmiana pozycje względem linii. Człon różniczkujący tłumi oscylacje robota.

Zasada działania:

1
Na początku pobierany jest odczyt z czujników i przetwożenie ich wartości w trybie DMA

2
Następnie na podstawie przypisanej wagi każdemu z czujników obliczany jest uchyb regulacji, od którego zależy prędkość silników

3
W kolejnym etapie podawana są nowe prędkości na każdy silnik

4
Proces powtarza się tak z częstotliwościa 100Hz

Przy zerowym błędzie oba silniki mają taką samą ustaloną prędkość. W zależności od błędu jeden z silników będzie przyśpieszał a drugi zwalniał. Przy bardzo ostrym zakręci jeden z silnikó dostanie większą moc a drugi będzie się kręcił w przeciwną stronę. Ważnym elementem algorytmu jest zapamiętywanie błędu i ciągła ich aktualizacja. Wypadnięcie z trasy skutkować będzie zwiększonym uchybem. Wartości czujników są przypisane liniowo, natomiast skajne czujniki otrzymały większe wartości.

3 Podsumowanie

W dokumencie opisano proces budowy robota klasy Line-Follower. Prace nad nim trwały nie całe 8 tygodni. Cel został osiągnięty, robot pojechał na zawodach Robotic Arena 2016. Dzień przed zawodami robot ostro bronił się przed wgraniem czegokolwiek, np. zaświecenie diodą. Podczas zawodów nie sprawiał żadnych problemów, program z poprawkami był wgrywany wielokrotnie. Przyczyny odmowy wgrania programu dzień przed zawodami są nieznane do dziś dzień. Natomiast godzinę przed rozpoczęciem naszej konkurencji spalił się stabilizator a później tranzystor. (Na szczęście w moim zespole był szaman lutowania, 10 minut później robot ozdrowiał). Mjølner poradził sobie bardzo dobrze, w klasie Line-Follower Light uzyskał 9 miejsce. Podsumowując, robot okazał się dobrą kontrukcja i po dwóch miesiącach starań pojechał

na zawodach. Jest jednak jeszcze wiele do zrobienia.

Cele na przyszłość:

- oprogramowanie modulu bluetooth
- oprogramowanie enkoderów
- ulepszenia algorytmu (człon całkujący)
- zapis programu strukturalnie, z podziałem na funkcje
- przetestowanie silników z przekładnią 10:1

4 Materiały źródłowe

Tutaj możemy napisać skąd czerpaliśmy wiedzę - np. z warsztatów, z Forbota, z płyt chodnikowych pod domem. Warto także wskazać na czym oparło się swoją pracę, np. tak [**?**].