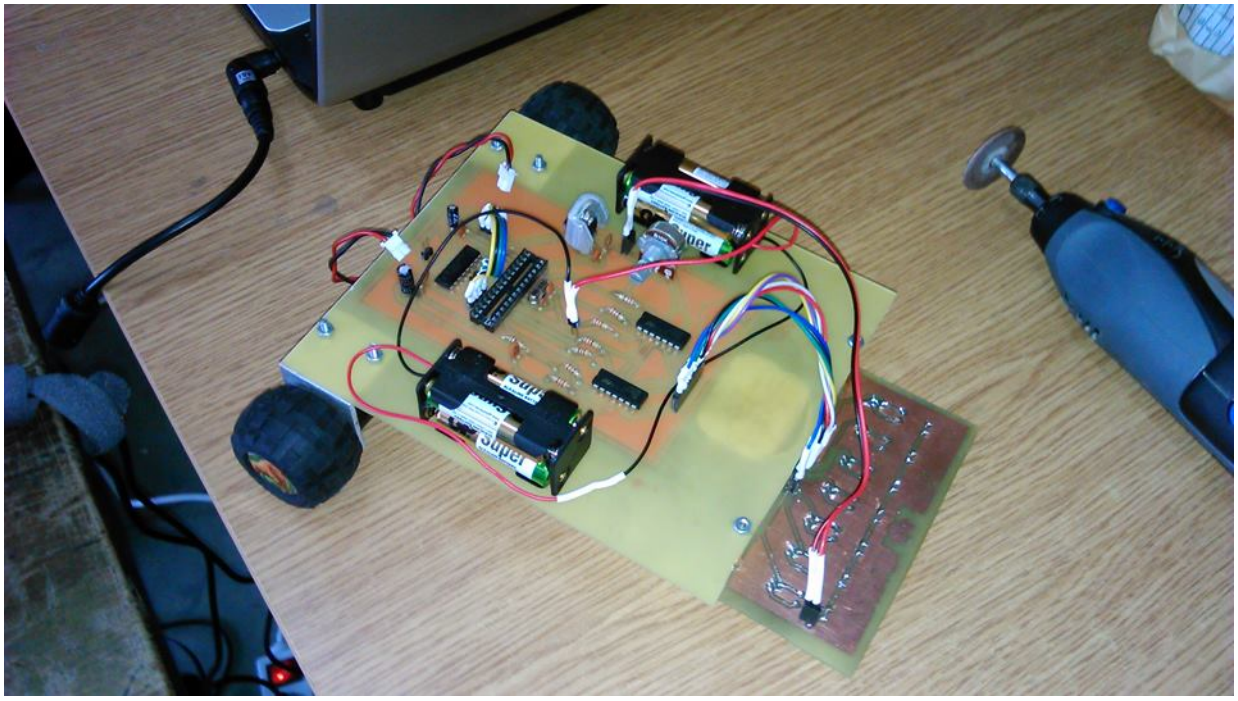
Dokumentacja robota typu line follower

Architektura Systemów Komputerowych

2013/2014

Nazwa robota:	
	Żubroń
Członkowie drużyny:	
mechanika, dokumentacja	Szymon Gramza
programowanie, projektowanie PCB	Przemysław Hoffmann
mechanika, wideo	Maciej Królikowski
lutowanie, wideo, dokumentacja	Damian Michalak
mechanika, programowanie, projektowanie PCB	Maciej Sobkowski
lutowanie, mechanika, wideo	Adam Szczesiak
mechanika, programowanie, testowanie	Kamil Wygralak
Prowadzący:	
	dr inż Rafał Klaus



Spis treści

1 Wstęp i opis projektu

Celem projektu było skonstruowanie robota typu Line Follower i wystartowanie w zawodach RoboDay 2014. Zadaniem robota typu Line Follower jest pokonanie trasy wyznaczonej przez czarną linię umieszczoną na jasnej powierzchni. Wykorzystując odpowiednie sensory i działając według zaimplementowanego algorytmu, robot musi utrzymać się na trasie przez 2 okrążenia. Wymagania dotyczące konstrukcji i specyfiki:

- musi mieścić się na kartce papieru formatu A4,
- waga nie jest ograniczona
- musi poruszać się w sposób autonomiczny, komunikacja z robotem w czasie przejazdu jest zabroniona,
- powinien być tak zaprojektowany, by można było go uruchomić na znak dany przez sędziego.

Wymagania dotyczące trasy:

- składa się z jasnej powierzchni i czarnej linii o szerokości 19± 1 mm,
- kąt zakrętu o promieniu 0 to maksymalnie 90 stopni,
- minimalna odległość między równoległymi liniami to 210mm,
- minimalna odległość między krawędziami planszy a torem to 210mm.

2 Idea rozwiązania

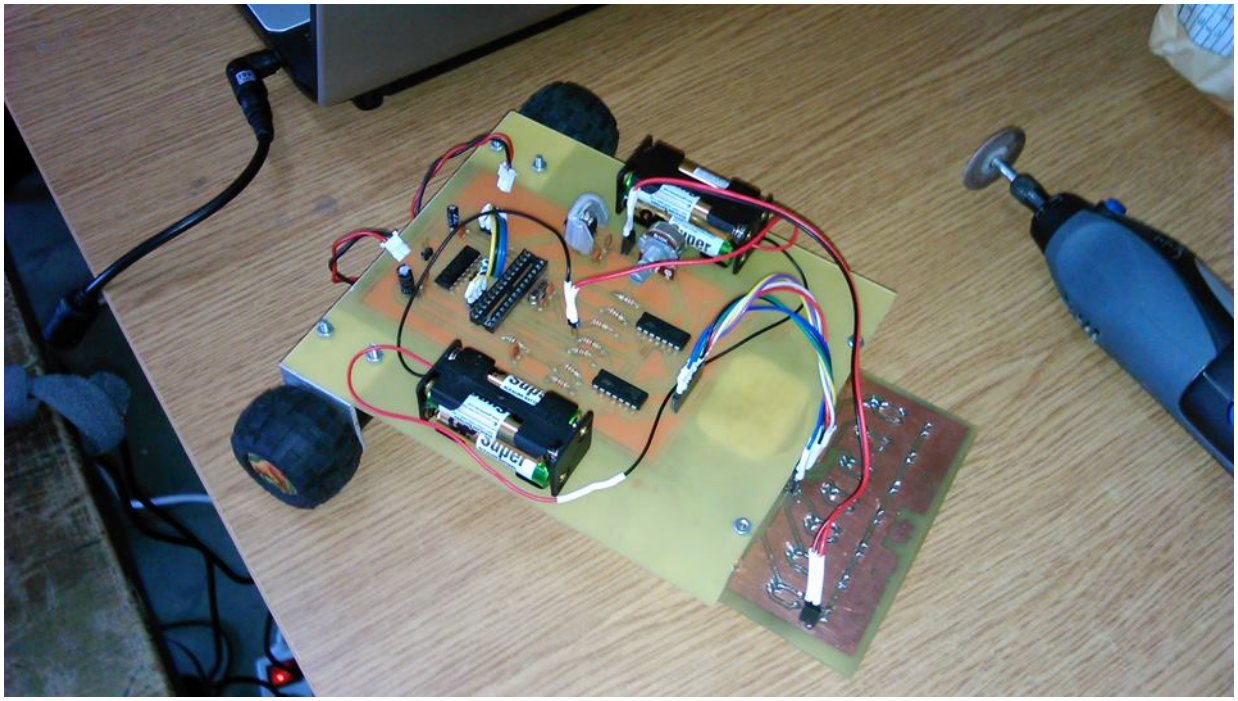
Stworzony robot spełnia wszystkie powyższe warunki. Autonomiczność robota uzyskano poprzez umieszczenie programu sterującego na mikrokontrolerze Atmega8. Do wykrywania trasy wykorzystano czujniki odbiciowe CNY70, które podłączone są do komparatorów. Następnie komparatory porównują te sygnały z napięciem odniesienia. Wyjścia komparatorów podłączone są do mikrokontrolera. Otrzymane sygnały stanowią dane wejściowe dla algorytmu. Mikrokontroler steruje pracą silników za pomocą mostka H. Robot posiada jedno źródło zasilania.

3 Mechanika

Głównymi komponentami mechanicznej strony robota są:

3.1 Konstrukcja nośna

Podstawą konstrukcji nośnej i jednocześnie miejscem montażowym elementów elektronicznych są dwie płytki laminatu szklano-epoksydowego typu FR4, jednostronnego o wymiarach odpowiednio: 143x169mm oraz 111x56mm. Większa płytka zwana dalej będzie płytką główną, zaś mniejsza zwana płytką czujnikową. Płytka czujnikowa została umieszczona z przodu konstrukcji i obniżona względem płytki głównej za pomocą 6 dystanserów(2x3 dystansery). Obniżenie uzyskane w ten sposób wyniosło 33mm i umożliwiło umieszczenie czujników odbiciowych w optymalnej odleości od podłoża 2,5mm. Ścieżki i miejsce przylutowania elementów płytki głównej zostały ulokowane w środkowej części tak, aby możliwe było wygodne przymocowanie



Rysunek 1: Zdjęcie konstrukcji

pozostałych podzespołów (baterii, silników itd). W wyniku wykonania powyższych czynności otrzymano sztywną konstrukcję mechaniczną. Początkowy projekt zakładał jedną płytkę montażową do której planowano zamocować za pomocą dystanserów płytki z układami elektronicznymi, jednak odrzucono ten pomysł mając na uwadze zmniejszenie całkowitej wagi robota. Zastosowany schemat konstrukcji pozwolił na łatwy dostęp do elementów, głównie do mikroprocesora, co było przydatne podczas programowania. Ponadto utrzymano transparentność oraz ułatwiono ewentualną wymianę podzespołów w przypadku awarii.

3.2 Układ jezdny

Układ jezdny składa się z części napędowej i mechanicznej. Część napędową tworzą dwa silniki 12V. Silniki umieszczono w tylnej części robota, mocując je za pomocą aluminiowych kątowników o wymiarach 30x60mm. W kątownikach nawiercono na środku otwory montownicze zgodne z notą katalogową silników.

Część mechaniczna składa się z dwóch elementów. Pierwszym są plastikowo-gumowe koła o wymiarach 43x28mm, pochodzące z zestawu klocków LEGO. Koła są zamontowane na silnikach za pomocą kleju na ciepło. Drugim elementem układu jezdnego jest zamocowane w przedniej części koło kastora, które zostało wykonane na tokarce w celu zmniejszenia kosztów projektu. Zapewnia ono stabilność oraz poziomuje konstrukcję, zmniejsza tarcie i zwiększa zwrotność robota. Dzięki temu zmniejszono także zużycie energii. Wypoziomowanie konstrukcji uzyskano umieszczając pomiędzy kołem kastora, a płytką drewniany klocek.

4 Elektrotechnika

4.1 Zasilanie

Zasilanie układu jest dostarczane z pakietu 8 baterii o napięciu 1,5V połączonych szeregowo co daje napięcie 12V. W układzie występują trzy różne napięcia:

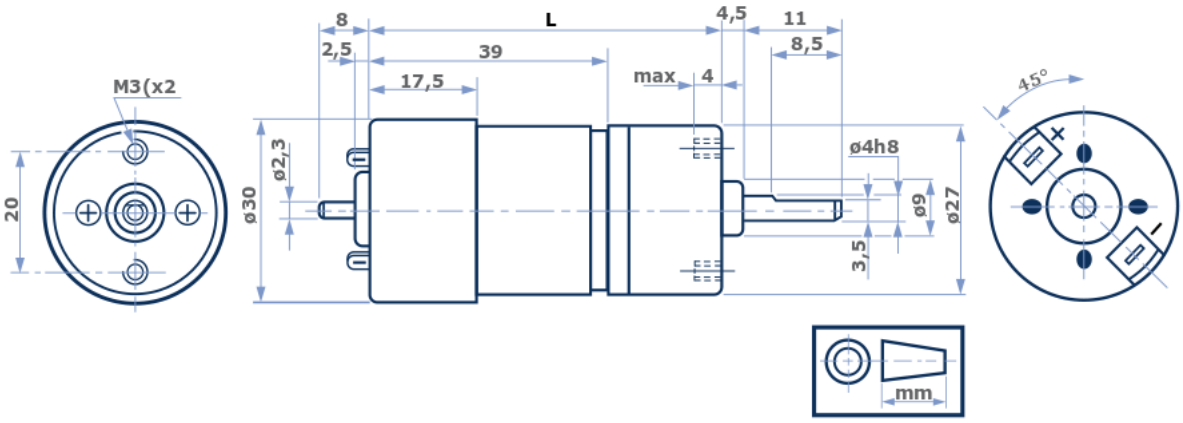
- 12V - napięcie silników Podawane na mostek H bezpośrednio z wyjścia baterii. Wykorzystywane jest do zasilania silników poprzez sterownik silników.
- 5V - napięcie zasilania logiki Jest to napięcie wyjściowe otrzymane z regulatora napięcia LM7805, który z wejściowego napięcia 12V daje na wyjściu 5V. Wykorzystywane jest ono do zasilania układów logicznych mikrokontrolera, mostka H i komparatorów oraz do zasilania diód w transoptorach.
- 0-5V - regulowane potencjometrem Wejściowym napięciem dla potencjometru jest 5V. Jest ono wykorzystywane do regulowania napięcia progowego, używanego do sterowania czułością transoptorów.

Szacuje się, że czas pracy na bateriach wynosi on około 2h +/-20 w zależności od jakości baterii.

Odrzucono wariant kombinacji komparatora z dodatkowymi resystorami, dzięki któremu miano uzyskać zjawisko histerezy (uniknięcia wielu zmian stanów 0→1 i 1→0 spowodowane podobnym co do wartości napięciom wejściowym komparatora pochodzących z transoptorów oraz potencjometru). Powodem odrzucenia było zbytnie skomplikowanie układu, dodatkowo stwierdzono, że problem który miała rozwiązywać można obejść stosując potencjiometr w którym można kontrolować napięcie odniesienia w zależności od warunków. Rozwiązano problem przegrzewania się stabilizatora przez zamocowanie przy nim radiatora oraz przez użycie rezystorów o większym oporze przy diodach CNY70.

4.2 Silniki

Robot napędzany jest dwoma silnikami firmy Micro motors o oznaczeniach producenta HL 149.6.10 AS. Silniki pracują w trybie ciągłej rotacji (continous rotation), umieszczono je w tylnej części konstrukcji. Do sterowania prędkością obrotów silników stosuje się sygnał PWM. Kierunek ich pracy jest ustalany programowo. Napięcie silników wynosi 12V. Ich wybór był motywowany brakiem konieczności stosowania dodatkowej przekładni gdyż mają wysoki moment obrotowy, ich wewnętrzne przełożenie wynosi 10:1. Dodatkowym czynnikiem była niska cena silników gdyż pochodziły z odzysku, jednak główną motywacją był wysoki współczynnik jakości do ceny.



Rysunek 2: Schemat silnika

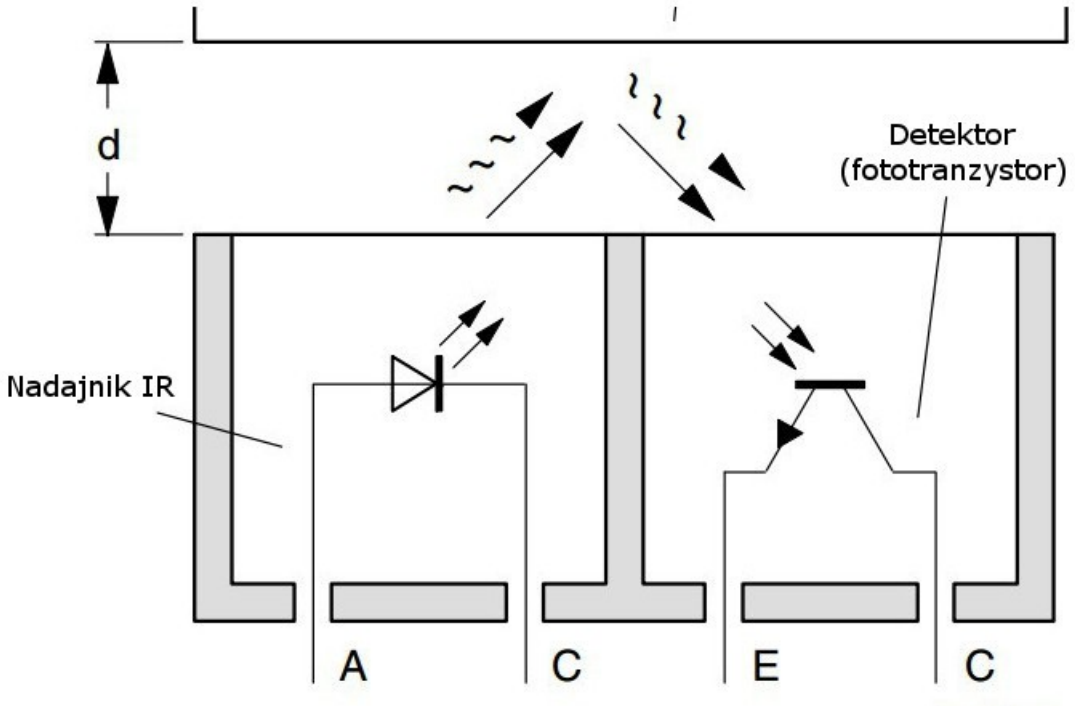
5 Elektronika

5.1 Czujniki

Do wykrywania trasy zastosowano transoptory CNY70 – układ diody podczerwonej i fototranzystora w jednej obudowie. Czujnik wysyła wiązkę promieniowania poprzez nadajnik podczerwieni, a następnie za pomocą fototranzystora, mierzy natężenie światła odbitego. Wyjściem jest sygnał napięciowy, zależny od natężenia światła padającego na ten detektor. Im więcej światła się odbije i dotrze do fotodetektora tym napięcie na wyjściu będzie miało wyższą wartość. Wykrycie czarnego koloru jest sygnalizowane stanem wysokim.

Specyfikacja czujników CNY70:

- Napięcie zasilania diody: 5 V
- Maksymalny prąd diody: 50 mA
- Maksymalne napięcie kolektor-emiter: 32 V
- Maksymalny prąd kolektora: 50 mA



Rysunek 3: Schemat działania transoptora

Do spolaryzowania napięcia na kolektorze fototranzystora wykorzystano rezystory podciągające do VCC o wartości 10kΩ. Wyjścia CNY70 podłączone są do pinów PD0-PD6 mikrokontrolera. Każda dioda połączona jest z rezystorami wartości 220Ω. Początkowo zastosowano rezystory 75Ω jednak powodowało to zbyt duży pobór prądu i skutkowało nadmiernym wydzielaniem się ciepła na regulatorze napięcia. Doświadczalnie dobrano resystory 220Ω. Czujniki rozmieszczone są od siebie w odległości 15mm co jest podyktowane regulaminową szerokością linii trasy. Doświadczalnie ustaliliśmy dystans między czujnikami na 75% szerokości linii. Odrzucono koncepcję zastosowania diód sygnalizujących stan czujników w celu oproszczenie schematu płytki. SCHEMAT PŁYTKI Z CZUJNIKAMI

5.2 Sterowanie silnikami

Do sterowania pracą silników wykorzystano gotowy mostek H. Jest to układ elektryczny umożliwiający sterowanie kierunkiem działania silników. Prędkość regulowana jest sygnałem PWM. Odrzucono wariant budowy własnego mostka H z tranzystorów z powodu nadmiernej komplikacji układu na płytce.

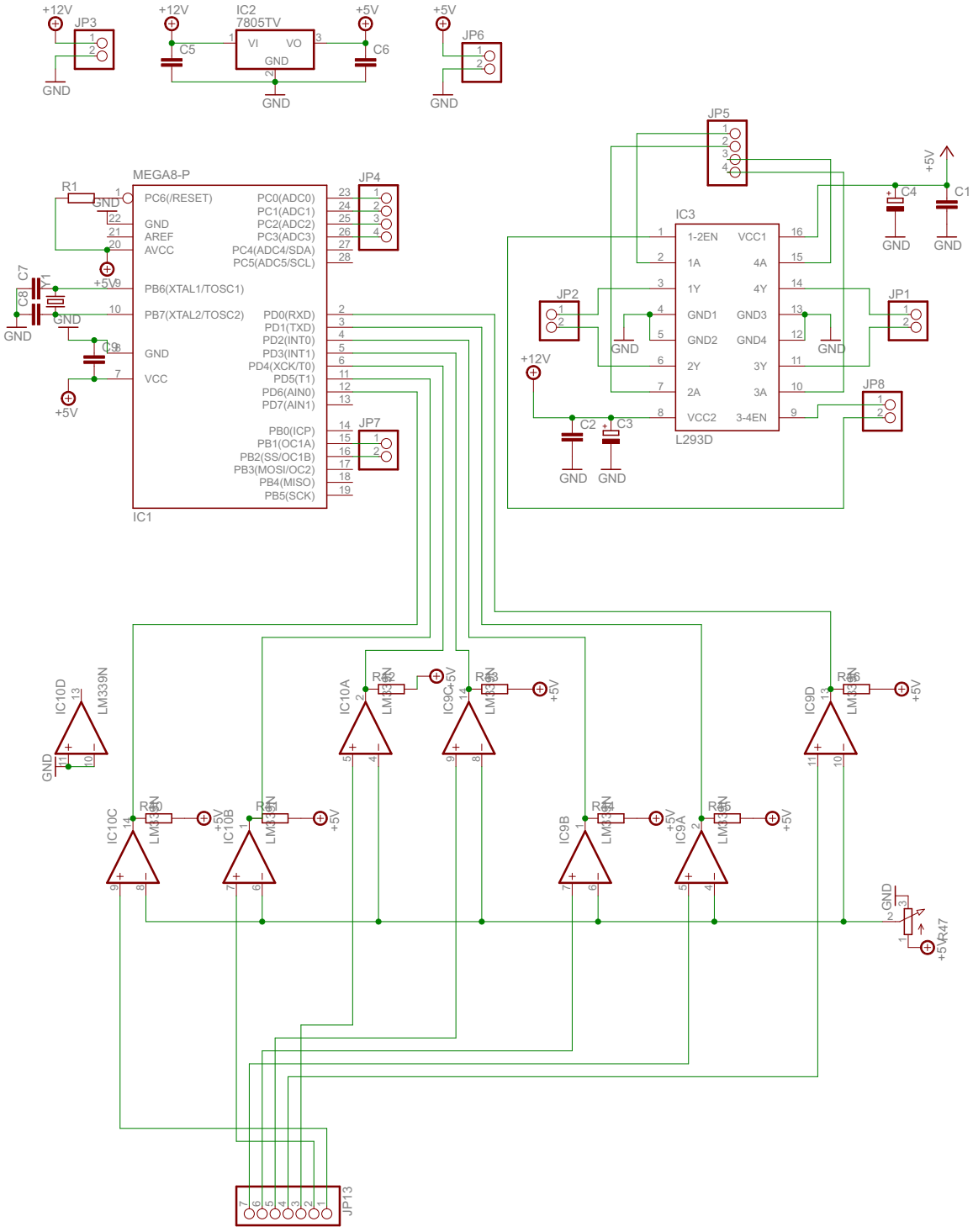
5.3 Płytka drukowana

Przygotowany robot składa się z dwóch płytek drukowanych: płytki głównej i płytki czujnikowej. Obydwie płytki są ze sobą połączone przewodami 7-żyłowym (każdy przenosi sygnał jednej czujki) oraz 2-żyłowym (zasilającym). Odrzucono wariant w którym cała elektronika znajdowałaby się na jednej płytce. Powodem był rozmiar silników jakie udało się nam uzyskać. Płytka główna jest na wysokości 41mm od podłoża. Aby zapewnić poprawne działanie czujek, muszą one znajdować się na wysokości do 5mm od podłoża. Sytuacja ta wymusiła przygotowanie drugiej płytki. Proces przygotowania płytki przebiegał następująco:

- przygotowanie schematu płytki w programie EAGLE
- wydrukowanie schematu płytki na papierze kredowym
- dokładne wyczyszczenie płytki po stronie pokrytej miedzią
- ścieranie papierem ściernym niewielkiej ilości miedzi z płytki
- podgrzanie płytki na żelazku przez ok. 30 sekund
- przyprasowanie wydrukowanego schematu płytki, pod wpływem temperatury toner ze schematu przykleja się do miedzi
- zanurzenie płytki w wodzie w celu odklejenia kartki ze schematem
- zanurzenie płytki w roztworze wody z wytrawiaczem na ok. 15 minut, w celu wytrawienia miedzi w miejscach na których nie ma tonera
- wytarcie pozostałego tonera octanem etylu

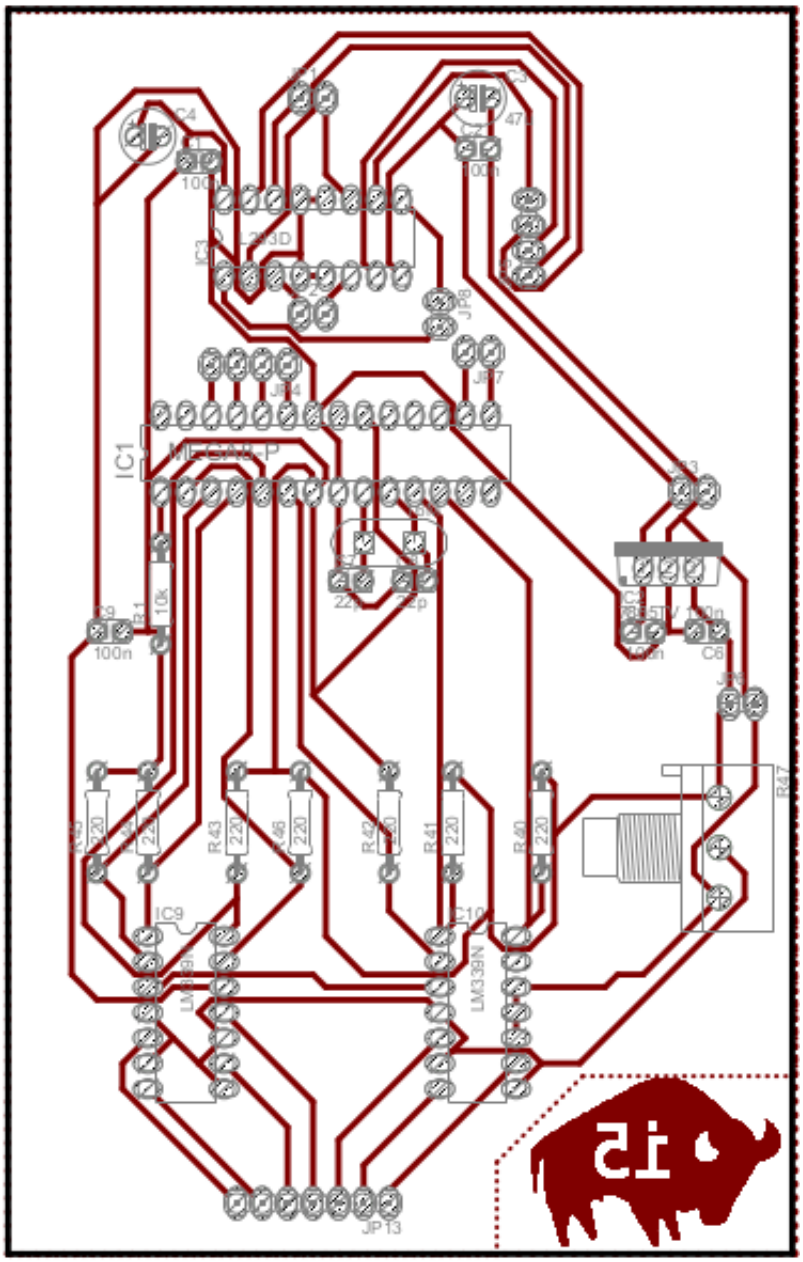
Cechy płytki głównej

- długość: 171mm
- szerokość: 143mm
- grubość ścieżek: 0,6096mm



9

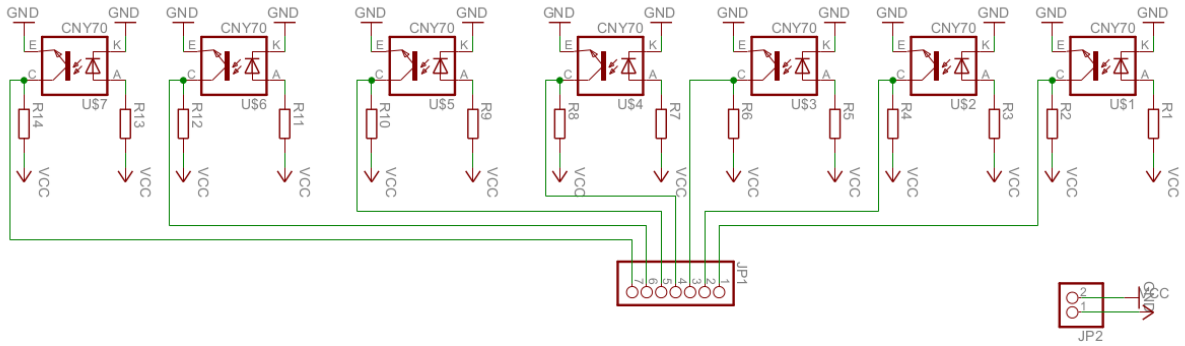
Rysunek 4: Schemat ideowy płytki głównej



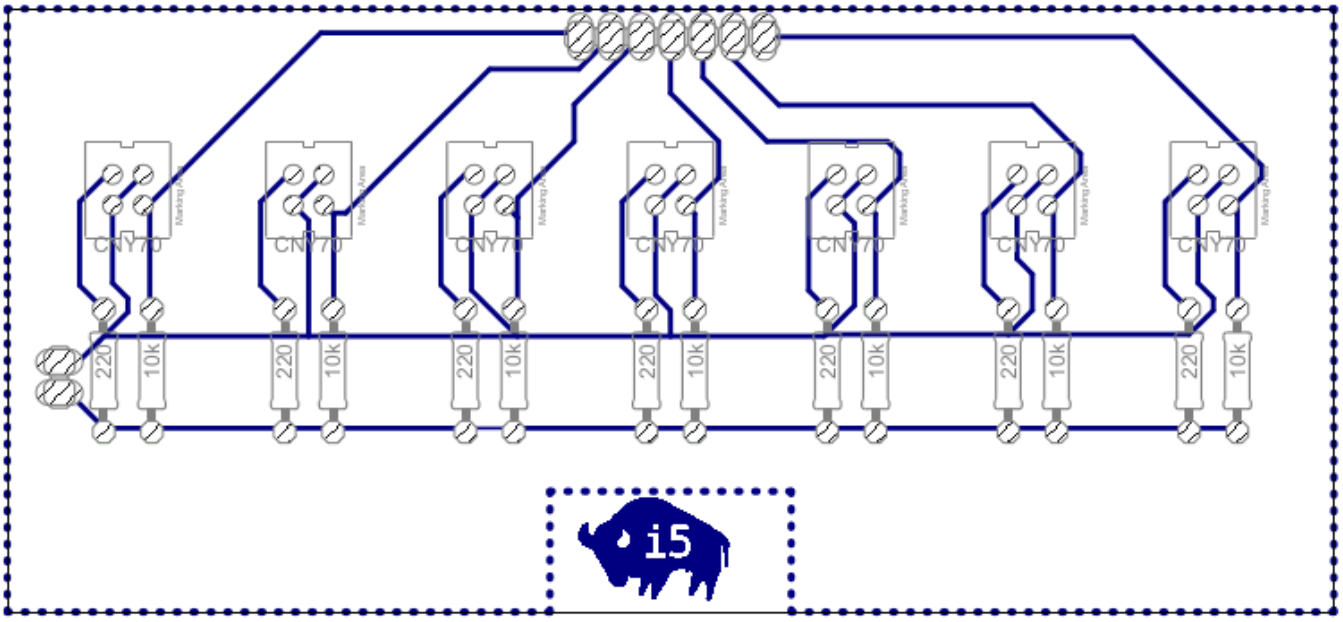
Rysunek 5: Schemat montażowy płytki głównej

Cechy płytki czujnikowej

- długość: 57mm
- szerokość: 111mm
- grubość ścieżek: 0,4064mm



Rysunek 6: Schemat ideowy płytki czujnikowej

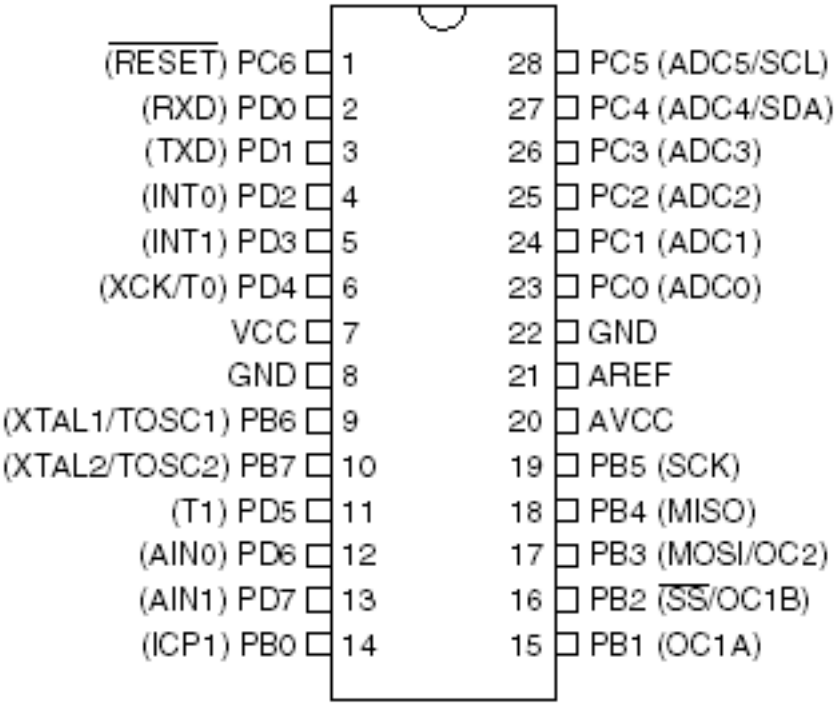


Rysunek 7: Schemat montażowy płytki czujnikowej

Największym problemem podczas przygotowanie płytki było odpowiednie dobranie czasu prasowania płytki z kartką na której był wydrukowany schemat. Po wielu próbach udało się dobrać odpowiedni - ok 5 minut. Czas ten zapewnił prawidłowe i całkowite przyklejenie się tonera do miedzi. Niestety podczas wytrawiania napotkano kolejny problem, mianowicie należało wyjąć płytkę z roztworu w odpowiednim momencie - tak, aby wytrawiła się miedź w miejscach na których nie ma tonera. Jednocześnie nie można trzymać płytki w roztworze za długo, gdyż zacznie się wytrawiać również część miedzi pod tonerem. W obawie przed nadmiernym wytrawieniem, wyjęto płytkę odrobinę za szybko. Później, podczas sprawdzania okazało się, że osobne ścieżki zwierają się ze sobą w kilku miejscach. Problem ten rozwiązano poprzez mechaniczne usunięcie miedzi śrubokrętem.

6 Mikrokontroler

Robotem steruje mikrokontroler ATmega8. Zasilany napięciem 5V, posiada 28 wyprowadzeń, których większość może pełnić różne role takie jak: wyjścia/wejścia cyfrowe, wejścia sygnałów przerwań, lub wejścia przetwornika ADC. Mikrokontroler taktowany jest zewnętrznym oscylatorem kwarcowym o wartości 16MHz. Do pinów 0-6(pin PD7 jest wolny) na porcie D, podłączone są wyjścia komparatorów. Piny od PC0 do PC3 służą do przyłączenia wyjść mostka H. Na pinach PB6-PB7 podłączony jest zewnętrzny oscylator.



Rysunek 8: Schemat mikrokontrolera

Wybraliśmy mikrokontroler Atmega8 z kilku powodów: głównym było wcześniejsze doświadczenie z pokrewnymi układami i związanymi z nim narzędziami programistycznymi. Przemawiało za nim także posiadanie przez nas programatora oraz niska cena układu. Atmega8 spełnia także wszystkie założone przez nas wymagania: ma odpowiednią ilość wyprowadzeń oraz timer który można skonfigurować w trybie PWM.

7 Oprogramowanie

7.1 Algorytm

Algorytm sterujący robotem został napisany w języku C. Następnie kod algorytmu został skompilowany w AVR-GCC (kompilator wydany przez firmę ATMEL). Skompilowany program został skopiowany do pamięci mikrokontrolera przy użyciu programu avrdude. Mikrokontroler jest połączony z wejściem USB komputera za pomocą programatora USB-ASP. Bazą algorytmu jest sterownik PID (kontroler proporcjonalno całkująco różniczkujący). Każdemu czujnikowi

przypisana jest waga (odpowiednio -3,-2,-1,0-1,2,3). Suma wag czujników, aktualnie wykrywających linię zapisana jest w zmiennej ERROR. W głównej pętli programu w pierwszym kroku sprawdzany jest stan czujników (obliczany jest ERROR). Jeśli ERROR jest różny od zera to uruchamiany jest PID. W przeciwnym razie (ERROR=0) możliwe są trzy warianty. Jeżeli żaden z czujników nie wykrywa trasy uruchamiana jest procedura odpowiedzialna za powrót na trasę. Odbywa się to poprzez odczytanie zapamiętanego stanu czujników z poprzedniego odczytu, co dostarcza informacji o prawidłowym kierunku powrotu. Pozostałe dwa warianty: wykrywanie przez wszystkie czujniki linii (skrzyżowanie) oraz stan wysoki jedynie na wyjściu środkowego transoptora skutkują podaniem na oba silniki równej wartości PWM.

7.2 Kod źródłowy

Listing 1: Kod

```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

#define START_PWM 150
#define TURN_PWM  140
#define MAX_PWM   255

#define SENSORS   7

#define K_P 15
#define K_I 2
#define K_D 1

void setup_motors();
void setup_sensors();

void motors_right();
void motors_left();
void motors_straight();

int compute_pid(int e, int pe);
int compute_error();
int is_line_visible();

int sensors[] = {
  PD1, PD2, PD3, PD0, PD4, PD5, PD6 };

int weitgts[] = {
  -3, -2, -1, 0, 1, 2, 3 };

int main()
{

  int pid_output = 0;
  int error = 0, prev_error = 0;
  int left_pwm = START_PWM, right_pwm = START_PWM;

  setup_sensors();
  setup_motors();

  while(1) {
```

```
    error = compute_error();

    motors_straight();
    if(error == 0) {

      if(is_line_visible()){
        OCR1A = TURN_PWM;
        OCR1B = TURN_PWM;
      }
      else {
        if (prev_error > 0){
          motors_right();
          OCR1A = TURN_PWM;
          OCR1B = TURN_PWM;
        }
        else {
          motors_left();
          OCR1A = TURN_PWM;
          OCR1B = TURN_PWM;
        }
      }
    }

    else {

      pid_output = compute_pid(error, prev_error);

      left_pwm = START_PWM - pid_output;
      right_pwm = START_PWM + pid_output;

      if(left_pwm > MAX_PWM){
        right_pwm += MAX_PWM - left_pwm;
        if(right_pwm < 0)
          right_pwm = 0;
        left_pwm = MAX_PWM;
      }

      if(right_pwm > MAX_PWM){
        left_pwm = left_pwm + MAX_PWM - right_pwm;
        if(left_pwm<0) left_pwm = 0;
        right_pwm = MAX_PWM;
      }

      OCR1A = right_pwm;
      OCR1B = left_pwm;

    }

    if (is_line_visible()) prev_error = error;

    _delay_ms(2);

  }
}

/**
 * Zwraca 1 jesli przynajmniej jeden czujnik wykrywa trase
 * W przeciwnym razie 0
```

```
 **/
int is_line_visible() {
  int i;
  for(i = 0; i < SENSORS; i++) {
    if(PIND & _BV(sensors[i]))
        return 1;
  }
  return 0;
}


/**
 * Zwraca blad obliczony na podstawie tablicy wag czujnikow
 * (Im bardziej skrajny czujnik tym ęwiksza waga)
 **/
int compute_error() {

  int error = 0;
  int i;
  for(i = 0; i < SENSORS; i++) {
    if(PIND & _BV(sensors[i]))
        error += weitgts[i];
  }

  return error;
}

/**
 * Zwraca sumaryczna korekte ustalona przez kontroler PID
 * (wagi czlonow to odpowiednio K_P, K_I i K_D)
 **/
int compute_pid(int e, int pe) {
    int p,d;
    static int i;
    p = e;
    i += e;
    d = e - pe;
    return (K_P * p) + (K_I * i) + (K_D * d);
}

/**
 * Ustawia piny odpowiedzialne za obsluge mostka H
 * (w tym takze ustawienie timera 1 w tryb Fast PWM)
 **/
void setup_motors() {

  DDRC |= _BV(PC0);
  DDRC |= _BV(PC1);
  DDRC |= _BV(PC2);
  DDRC |= _BV(PC3);


  TCCR1A = (_BV(COM1A1) | _BV(COM1B1));
  OCR1A = START_PWM;
  OCR1B = START_PWM;
  TCCR1A |= _BV(WGM10);
  TCCR1B = _BV(WGM12) | _BV(CS10);
  DDRB |= (_BV(PB1) | _BV(PB2));
```

```
}

/**
 * Ustawia port do ktorego przylaczone sa sygnaly z czujnikow
 * jako wejsciowy
 **/
void setup_sensors() {

  DDRD = 0x00;

}

/**
 * Ustawia sygnaly na pinach sterujacych kierunkiem silnikow
 * tak, aby robot skrecil w lewo
 **/
void motors_left() {

  PORTC &= ~_BV(PC0);
  PORTC |= _BV(PC1);
  PORTC |= _BV(PC2);
  PORTC &= ~_BV(PC3);

}

/**
 * Ustawia sygnaly na pinach sterujacych kierunkiem silnikow
 * tak, aby robot skrecil w prawo
 **/
void motors_right() {

  PORTC |= _BV(PC0);
  PORTC &= ~_BV(PC1);
  PORTC &= ~_BV(PC2);
  PORTC |= _BV(PC3);
}

/**
 * Ustawia sygnaly na pinach sterujacych kierunkiem silnikow
 * tak, aby robot jechal prosto
 **/
void motors_straight() {

  PORTC &= ~_BV(PC0);
  PORTC |= _BV(PC1);
  PORTC &= ~_BV(PC2);
  PORTC |= _BV(PC3);

}
```

8 Serwisowania i konserwacja

Gniazdo wyjściowe płytki czujników stanowi jednocześnie punkt pomiarowy umożliwiający sprawdzenie poprawności działania czujników. Serwisowalne elementy robota:

- silniki
- koła (wymienne ogumienie)

- baterie
- płytka czujników
- mikrokontroler (wymienny na dowolny kompatybilny z Atmega8)
- przewody łączące płytki

9 Inżynieria oprogramowania i metodyki prowadzenia projektu

9.1 Analiza SWOT członków zespołu

Na podstawie obserwacji sporządzono listę mocnych i słabych stron poszczególnych członków drużyny:

Członkowie drużyny:	mocne	słabe
Maciej Sobkowski	doświadczenie z elektroniką talent do przekazywania wiedzy	brak cierpliwośći brak precyzyjnych zdolności manualnych
Damian Michalak	umiejętność rozwiązywania problemów praca w grupie	spóźnialstwo nie przepadam za programowaniem w niskim poziomie
Przemysław Hoffmann	organizacja spotkań/mobilizowanie grupy zdolność do wchłaniania wiedzy, szczególnie o C	wywoływanie konfliktów w drużynie prawie zerowy poziom wiedzy elektronicznej
Szymon Gramza	skrupulatność i dokładność zamiłowanie do majsterkowania	niski poziom skupienia niski poziom wiedzy elektronicznej
Adam Szczesiak	precyzyjność punktualność	brak szczegółowej wiedzy elektronicznej brak zamiłowania do języka C
Kamil Wygralak	zamiłowanie do majsterowania koordynacja pracy grupy	łatwość do rozkojarzania się szybkie zniechęcanie się gdy coś nie idzie
Maciej Królikowski	umiejętność organizacji pracy ambicja	małe umiejętności manualne problemy z koncentracją

9.2 Metodyka

Do zarządzania projektem wykorzystano metodykę PRINCE2 (Projects In Controlled Environments). Jest ona stosowana do zarządzania i sterowania projektami wszelkiego rodzaju i wszelkiej wielkości. Głowna idea polega na optymalnym wykorzystywaniu zasobów, które według PRINCE2 dzielą się na trzy rodzaje: pieniądze, ludzie i sprzęt. PRINCE2 cechuje podejście procesowe do zarządzania projektem. Definiuje szczegółowo siedem procesów najwyższego rzędu:

- Przygotowanie założeń projektu celem tego procesu jest przygotowanie projektu do uruchomienia. Polega na wybraniu kierownika projektu oraz ustaleniu podziału pracy między osoby lub zespoły
- Strategiczne zarządzanie projektem bierzące zarządzanie projektem przez kierownika projektu na podstawie raportów okresowych o jego stanie. Proces ten trwa przez cały czas trwania projektu.
- Inicjowanie projektu obejmuje planowanie zarządzania zasobami, podziału zajęć między drużyny oraz budowanie harmonogramu. Podczas tego procesu należy mieć na uwadze czas zakończenia projektu i umiejętne operować czasem.
- Sterowanie etapem Projekty realizowane według metodyki PRINCE2 są podzielone na etapy zarządcze. Dokładna liczba etapów nie jest określona jednak niektóre z nich są wymagane, jak np. zgoda na wykonywanie grupy zadań, ocena postępów czy raportowanie o ważnych zdarzeń.
- Zarządzanie wytwarzaniem produktów PRINCE2 to metodyka oparta na produktach. Produktem może być rzecz materialna lub niematerialna. W tym procesie ważna jest komunikacja między kierownikiem projektu a zespołami wykonującymi produkty.
- Zarządzanie zakresem etapu Zgodnie z PRINCE2 każdy etap musi być ukończony i zaakceptowany zanim kierownik projektu autoryzuje przejście do następnego etapu. W tym procesie weryfikowane jest, czy etap dostarczył wszystkie wymagane produkty i czy może rozpocząć wykonywanie następnego.
- Zakończenie projektu Według metodyki PRINCE2 projekty muszą być zamykane w sposób uporządkowany i kontrolowany. Wszystkie doświadczenia zdobyte w trakcie prowadzenia projektu są rejestrowane, tworzony jest dokument przekazania i planowany jest przegląd powdrożeniowy.

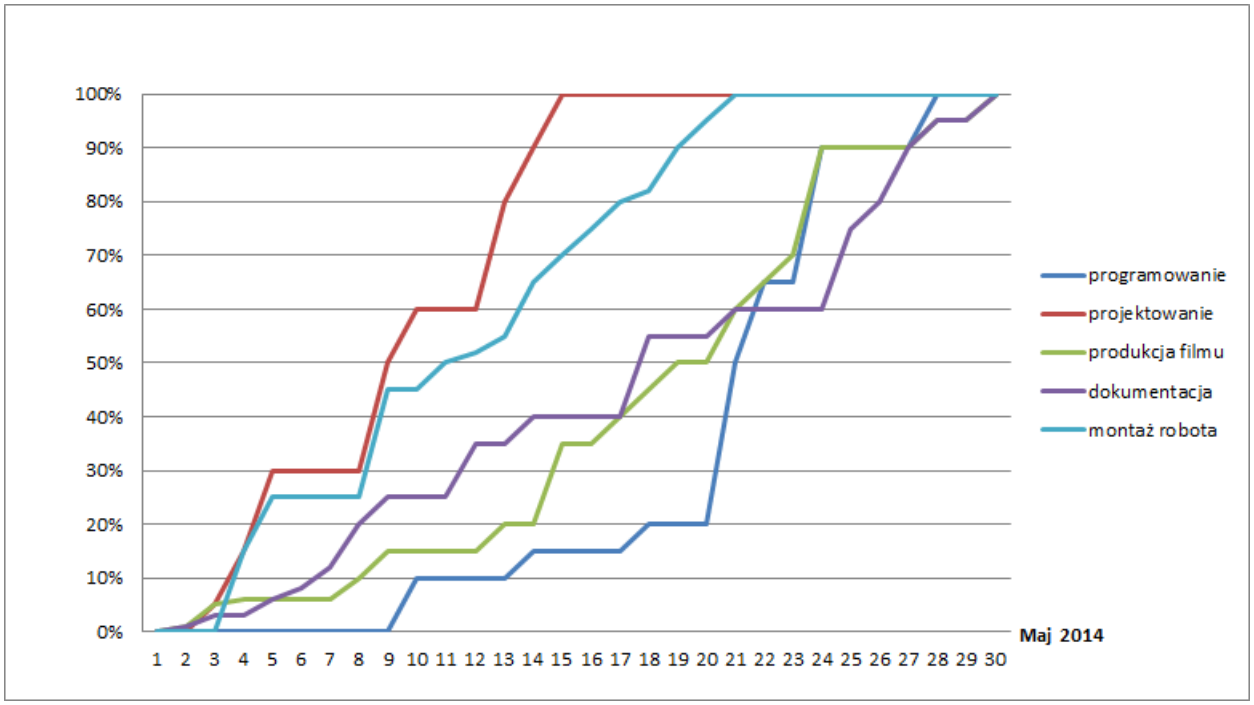
9.3 Kosztorys

Niezbędne elementy oraz koszt ich zakupu przedstawia poniższa tabela:

Element	Ilość	[PLN]/szt	Suma
SILNIK	2	5,00	10,99
LISTWA ASS02029Z	1	3,00	3,00
LISTWA KOŁK. 1x40pin PROSTA r=2,54mm	1	0,70	0,70
PINHEADER-GNIAZDO 40p PROSTY r=2,54mm	1	1,21	1,21
KOND. CERAMICZNY 22pF	10	0,08	0,80
GNIAZDO BH-10S 10pin PROSTE	50	0,01	0,50
REZ. WĘGL. 0,25W -1K 5%	0,1	5,01	0,50
LED 3mm ZIELONA-DYFUZYJNA 50 mcd	0,01	15,01	0,15
STAB. 7805 5V/1.5A T0220	1	1,21	1,21
GNIAZDO DC 5,5/2,1 KĄT. W DRUK-2020	1	1,00	1,00
ZŁĄCZE AK500/2 r=5mm TAJWAN-NIEB.	1	0,70	0,70
JUMPER r=2,54mm ZAMKNĘTY h=6mm	0,04	5,01	0,20
DŁAWIK 10μH 160mA 2,5R OSIOWY	0,10	30,00	0,30
DIODA PROST. 1N4007 1A/1000V	10	0,01	0,10
LAMINAT JEDNOSTR. 12x21,5mm GR=1,5m	1	5,81	5,81
TRANSOPTOR	4	2,80	11,20
ATMEGA8A-PU 8kb-FL 1kb-RAM 512b	1	7,50	7,50
NADSIARCZAN SODU-B327 100g	1	4,50	4,50
WIERTŁO 0.8mm	5	1,80	9,00
WIERTŁO 1.0mm	2	1,00	1,00
WIERTŁO 1.2mm SARIUS	3	1,00	3,00
KOND. EL. 4,7μF/25V 85 C	10	0,05	0,50
KOND. KEX47μF/25V 105 C	20	0,05	1,00
KOND. EL. 10μF/25V 5x11mm 105 C	15	0,05	0,75
KOND. EL. 100μF/25V 5x11mm 105 C	30	0,05	1,50
KOND. EL. 100μF/50V 8x12mm 105 C	40	0,10	0,40
KOND. EL. 470μF/50V 13x25mm	1	0,60	0,60
REZ. WĘGL. 0,25W -75R 5%	0,1	5,00	0,50
REZ. WĘGL. 0,25W -330R 5%	0,1	5,00	0,50
REZ. WĘGL. 0,25W -470R 5%	0,1	5,00	0,50
REZ. WĘGL. 0,25W -1K 5%	0,1	5,00	0,50
REZ. WĘGL. 0,25W -2,2K 5%	0,1	5,00	0,50
REZ. WĘGL. 0,25W -1K 5%	0,1	5,00	0,50
REZ. WĘGL. 0,25W -47K 5%	0,1	5,00	0,50
MIKRO/SW 6x6mm h=1,5	0,1	30,0	3,00
STAB. 7805 5V/1,5A T0220	2	1,20	2,40
KOND. CERAM. 100N/50V/105 r=2,54mm	0,50	10,00	0,50
DŁAWIK 10μh 160mA 2,5R OSIOWY	0,02	30,00	0,60
LAMINAT JEDNOSTR.19x31 1mm EPOXYD	1	9,00	9,00
SŁUPEK DYST.M3x20mm GWINT Z/W-NIKL.	4	0,75	3,00
NAKRĘTKA M3 -CYNOWANA	0,04	7,00	0,28
ŚRUBA M3 GRZYBEK 10mm	0,40	30,00	1,20
LM 339N DIP14	4	0,80	3,20
L298N POWER DRIVER SQL15	1	11,00	11,00
RURA TERM. RT 2,4/1,2	1	0,80	0,80
L293DNE	1	6,00	6,00
TULEJKA DYST. DR3 6/25mm PLASTIK	0,02	20,00	0,40
POTENCJ. OBROT. B10K LINIOWY L=15mm	1	3,00	3,00

BATERIA 1,5V	8	1,50	12,00
WYTRAWIACZ	1	4,99	4,99
KOSZYCZKI NA BATERIE	2	2,00	4,00
CZUJNIKI CNY70	5	2,80	14,00
CZUJNIKI CNY70	3	4,80	14,40

9.4 Czas pracy



Rysunek 9: Wykres postępu

9.5 Narzędzie i elementy

Opis narzędzi i elementów użytych przy realizacji projektu.

9.6 Narzędzia

Lista narzędzi:

- stacja lutownicza
- multiszlifierka precyzyjna z zestawem wymiennych końcówek
- zestaw śrubokrętów precyzyjnych
- zestaw cążków i kombinerek precyzyjnych
- latarka czołowa
- pistolet do kleju na ciepło
- miernik uniwersalny
- ładowarka do akumulatorków
- płytka prototypowa

Wyżej wymienione narzędzia umożliwiły zamocowanie, unieruchomienie podzespołów na płytkach pcb oraz stworzenie zwartej konstrukcji robota. Ponadto umożliwiły m.in. wiercenie, szlifowanie, matowienie powierzchni, przygotowanie elementów konstrukcyjnych własnej roboty takich jak dystanser przedniego koła w celu wypoziomowania konstrukcji.

10 Uwagi końcowe

W kwestii elementów, które można by poprawić, należy wymienić:

- użycie programu CAD do zaprojektowania robota
- wzmocnienie konstrukcji, gdyż zauważono wyginanie się płytki spowodowane wagą silników
- zminiaturyzowanie układu elektronicznego

11 Spisy

11.1 Spis rysunków i tabel

Spis rysunków

11.2 Spis zawartości DVD

- DTR w pdf
- katalogi elementów elektroniki, elektrotechniki, mikrokontrolera
- wideoklip
- niezbędne kodeki do wideoklipu
- prezentacja PowerPoint
- wydruki do termotransferu

12 Literatura

Literatura pomocna przy tworzeniu robota:

- "Język C dla mikrokontrolerów AVR. Od podstaw do zaawansowanych aplikacji". Tomasz Francuz.

Linki do not katalogowych:

- CNY70
- Atmega8

- Silniki
- Sterownik silników
- AVR221