Uniwersytet Śląski
Wydział Informatyki i Nauki o Materiałach

Grzegorz Grzyk
206778

PRACA LICENCJACKA

Sterowanie podwoziem

Promotor: dr Ireneusz Gościniak

Sosnowiec, 2007

Spis treści

1. Wprowadzenie

1.1.Wstęp

W celu opracowania tematu sterowania podwoziem należy w pierwszej kolejności poznać specyfikę problemu. Wiąże się to z koniecznością zapoznania się podstawowymi pojęciami związanymi z przedstawionym tematem.

STEROWANIE - oddziaływanie na określony układ, służące do zapewnienia jego zachowania się w żądany sposób. Układ dynamiczny, w którym można wymusić określone przebiegi procesu za pomocą oddziaływań sterujących jest nazywany obiektem sterowania. Obiekt ten można opisać za pomocą współrzędnych jego stanu, parametru oraz struktury powiązań między elementami. Na wejściu obiektu sterowania działają wielkości użyteczne (sygnały sterujące) i zakłócające (szumy), na wyjściu otrzymuje się sygnały sterowane. (...) W systemach technicznych sterowanie polega na oddziaływaniu na strumienie energii lub materiałów. Często stosuje się sterowanie zdalne, w którym sygnały sterujące są przesyłane do obiektów sterowania znajdujących się w znacznej odległości od urządzenia sterującego. Rozróżnia się sterowanie ręczne i automatyczne [1].

STEROWANIE AUTOMATYCZNE - realizowane przez automatyczne urządzenie sterujące, którego sygnały wyjściowe (sygnały sterujące) oddziałują na obiekt. Zespół funkcjonalny złożony z obiektu oraz urządzenia sterującego (np. układ mikroprocesorowy, odpowiednio zaprogramowany komputer) tworzy układ sterowania. (...) Układ sterowania może być zamknięty - jeśli sygnały wyjściowe obiektu oddziałują zwrotnie na urządzenie sterujące, lub otwarty - jeśli urządzenie sterujące nie korzysta z bieżących informacji o sygnałach sterowanych, a wpływ na wielkość sterowaną odbywa się według programu określonego przez sygnał zadany. (...) Ważnym rodzajem sterowania automatycznego jest sterowanie numeryczne (cyfrowe) realizowane za pomocą komputera, w numerycznych układach sterowania są wykorzystywane układy logiczne [1].

SYGNAŁ - to abstrakcyjny model dowolnej mierzalnej wielkości zmieniającej się w czasie, generowanej przez zjawiska fizyczne lub systemy. Ponieważ sygnał niesie informację o naturze badanych zjawisk lub systemów, w niektórych dziedzinach nauk jest on traktowany jak nośnik informacji. Sygnał oznacza zatem przepływ strumienia informacji, przy czym przepływ może odbywać się w jednym lub w wielu wymiarach [2].

PODWOZIE - (motoryzacja), zespół elementów pojazdu, które odpowiadają za przeniesienie napędu z silnika na koła oraz za zachowanie się pojazdu na drodze. W skład podwozia wchodzi: układ hamulcowy, układ kierowniczy, układ napędowy, zawieszenie. Podwozie obok nadwozia (i silnika) jest częścią składową pojazdu (samobieżnego) [2].

Zatem obiektem naszego sterowania staje się podwozie, które wymaga rozbudowania o mechanizmy napędzające elementy podwozia. Mechanizmy te maja za zadanie zamianę odbieranych sygnałów cyfrowych na ruch odpowiednich części podwozia. Tak skonstruowany pojazd kołowy staje się obiektem sterowania. Używając terminu sygnały cyfrowe zakładamy, że sterowanie będzie odbywało się w sposób otwarty i automatyczny, według wyżej podanych definicji. Zadanie wykonania sterowania automatycznego wiąże się z opracowaniem i zbudowaniem urządzenia sterującego. Dochodząc do etapu kiedy posiadamy dwa urządzenia, obiekt sterowania i sterownik, stajemy przed problemem połączenia urządzeń. Połączenie to powinno zapewnić możliwość przesyłania sygnałów sterujących pomiędzy urządzeniami. W końcowym etapie powinniśmy uzyskać kompleksowy system sterowania podwoziem.

1.2. Cel i zakres pracy

Podstawowym celem pracy było opracowanie projektów i wykonanie układów pozwalających w jak najbardziej kompleksowy sposób sterować modelem pojazdu. Zadanie wiązało się z wyposażeniem podwozia w układ napędowy, układ kierowniczy oraz system elektroniczny. Koniecznym było opracowanie sterownika pojazdu, który pozwoliłby na precyzyjne sterowanie pojazdem. Pojazd wraz ze sterownikiem powinien tworzyć system otwarty, zapewniając użytkownikowi sprawowanie pełnej kontroli nad pojazdem. W celu zapewnienia pełnej kontroli oraz precyzyjności sterowania sterownik powinien stanowić komputer wyposażony w analogowy manipulator.

Proces opracowania systemu sterowania został podzielony na dwie części. Pierwsza część stanowi zdefiniowanie celów, które należy osiągnąć podczas budowy oraz uruchomienia pojazdu i sterownika. Drugą część stanowią zagadnienia dotyczące rozbudowy podwozia o mikrokomputer ARM oraz zagadnienia dotyczące sterowania systemu z stanowiska komputerowego. Pierwszą część pracy podzielono na szereg mniejszych zagadnień, mniej skomplikowanych do opracowania. Dopiero opracowanie każdego zagadnienia zaowocuje spełnieniem głównego celu pracy.

Lista zagadnień stanowiących zakres pracy:

- Zaopatrzenie podwozia w odpowiedni napęd i układ kierowniczy.
- Opracowanie metody sterowania prędkością pojazdu i wychyleniem kół.
- Opracowanie sterownika pozwalającego w pełni kontrolować ruchy pojazdu.
- Zbudowanie urządzeń z wykorzystaniem techniki cyfrowej.
- Opracowanie i wykonanie komunikacji pomiędzy sterownikiem a pojazdem.
- Zapewnienie autonomicznego zasilania pojazdu.
- Rozbudowanie systemu o układ mikrokomputera opartego na procesorze ARM.
- Zapewnienie możliwości dalszej rozbudowy pojazdu.

Pierwszy cel wynika stąd, że podwozie nie zawierało napędu i należało sprawić by pojazd poruszał się sprawnie, to znaczy posiadał dość duże przyśpieszenie, prędkość. Następne cztery punkty można było zrealizować stosując gotowe moduły i aparaturę sterującą jednak postanowiono odrzucić takie rozwiązanie ze względu na brak możliwości poznania dokładnej budowy i działania takich układów. Kolejny punkt wymaga nieco dokładniejszego omówienia, gdyż używając określenia mikrokomputer niemiano na myśli, podstawowego układu zawierającego mikroprocesor 32-bitowy ARM. Używając nazwy mikrokomputer podkreśla się fakt iż układ miała stanowić płytka rozwojowa zbudowana na bazie procesora ARM wyposażonego w szereg peryferii komputerowych takich jak: karta sieciowa, kontrolery magistrali USB, czytniki kart pamięci itp. oraz wyposażenia układu w pełni sprawny system operacyjny, w tym przypadku linux. Zagadnienie mikrokomputera ARM, wraz ze wszystkimi elementami z nim związanymi, zostanie kompleksowo opisane w oddzielnym rozdziale. Ostatni punkt zapewnia tak zwaną otwartą architekturę, co wiązało się z zachowaniem standardów wejściowych oraz wyjściowych w budowanych układach. Dlatego w przyszłości zbędne stanie się korzystanie z dodatkowych układów dopasowujących standardy komunikacji pomiędzy układami.

2. Idea budowy mobilnej platformy

Idea przyświecająca budowie pojazdu była bardzo ambitna. Finalnie podwozie miało stać się szybkim pojazdem wyposażonym w wydajny uniwersalny układ elektroniczny oparty na architekturze ARM, z zainstalowanym i w pełni funkcjonalnym systemem Linux. Orientację w otoczeniu miała zapewniać kamera z możliwością obrotu w dwóch płaszczyznach, w połączeniu z zestawem czujników odległości i systemem pozycjonującym. Dwukierunkowa szerokopasmowa komunikacja pojazdu z komputerem miała odbywać się torem transmisji radiowej wykorzystując pasmo radiowe 2,4Ghz. Komunikacja miała być zapewniona przy pomocy urządzeń radiowych wykorzystywanych do budowy komputerowych sieci bezprzewodowych WLAN. Całe sterowanie miało odbywać się po stronie komputera przy wykorzystaniu analogowego manipulatora podpiętego do portu komputera wspomaganego rozbudowanymi algorytmami autonomicznego sterowania. Jednak ze względu na obszerność całego zagadnienia i ograniczony czas przeznaczony na budowę, wykonanie autonomicznego robota okazało się niemożliwe. Należało więc zawęzić zakres prac tak by możliwe było ukończenie projektu w zadanym czasie.

Tak powstałe założenia projektu już na samym początku podkreśliły pewne rozwiązania i odrzuciły inne. W pierwszej kolejności należało dobrać i zamontować odpowiedni silnik. Kwestia doboru napędu długo pozostawała otwarta, ponieważ przy doborze należało wziąć pod uwagę wiele różnych parametrów i czynników. Zadanie doboru jednostki napędowej zostanie opisane w późniejszym rozdziale pracy. Następnie dokonano wyboru odpowiedniego serwomechanizmu posiadającego niewielkie wymiary, a zarazem dysponującego wystarczającym momentem obrotowym potrzebnym do skręcenia kół. Konieczne było zbudowanie układu bezpośrednio sterującego silnikiem i serwomechanizmem, który będzie przetwarzał odebrane dane i wysyłał odpowiednie impulsy elektryczne do silnika i serwomechanizmu. Najbardziej optymalnym rozwiązaniem było zbudowanie układu, którego centralnym elementem jest układ mikroprocesorowy posiadający możliwość komunikacji z zewnętrznymi źródłami danych oraz sterowania kolejnymi podzespołami. Optymalnym rozwiązaniem okazał się mikrokontroler jednoukładowy ze względu na dużą skalę integracji oraz integrację z różnego typu układami peryferyjnymi. Mając na względzie możliwości konstrukcyjne oraz kwestie montażu urządzenia, sterownik pojazdu zdecydowano się wykonać wykorzystując mechaniczne elementy analogowego

joysticka komputerowego, do których zostanie dołączony własnoręcznie wykonany układ elektroniczny. Na pewnym poziomie prac nad joystickiem okazało się, że koniecznym jest zbudowanie kolejnego układu, który przedstawia użytkownikowi w sposób dla niego zrozumiały, stopień wychylenia drążka. Następnie został opracowany system komunikacji między pojazdem a joystickiem. Zdecydowano że łączność będzie odbywać poprzez przewodowe medium transmisyjne wykorzystując szeregową transmisję danych. Taka metoda została wybrana ze względu na łatwość późniejszego zintegrowania z systemem mikrokomputera i stanowiska PC. Opracowując problem autonomicznego zasilania pojazdu, zdecydowano się na wykorzystanie bezobsługowych akumulatorów modelarskich, wraz z gotową ładowarką procesorową, która w krótkim czasie przywraca stan naładowania akumulatorów.

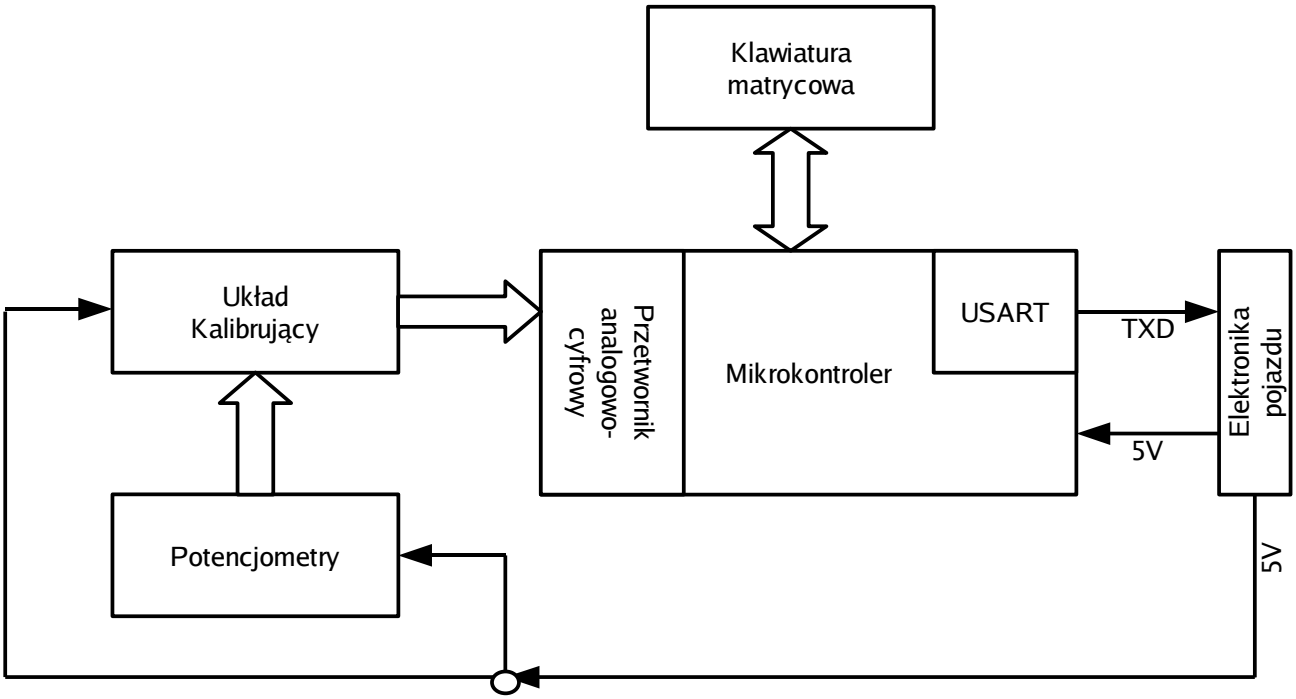
Istotna sprawą z poziomu idei jak i założeń projektu była koncepcja dalszej rozbudowy systemu sterowania o możliwość zdalnego sterowania z komputera, każdy układ został zbudowany w postaci modułu. Zdecydowano się także wykorzystać komputerowe standardy transmisji szeregowej tak by można każdy moduł bezpośrednio łączyć ze sobą, z mikrokomputerem bądź komputerem stacjonarnym.

2.1. Zasada działania

Całość systemu sterowania została podzielona na trzy moduły. Pierwszy moduł stanowi układ, który bezpośrednio steruje mechanizmami pojazdu. Drugi moduł stanowią układy elektryczne joysticka, które cały czas sprawdzają stopień wychylenia drążka oraz stan przewodzenia wszystkich przycisków. Trzeci moduł stanowi układ, którego zadaniem jest przedstawienie na wyświetlaczu diodowym zawartości paczek przesyłanych podczas transmisji danych między joystikiem a pojazdem. Sercem każdego z modułów jest mikrokontroler, który każdemu z nich zapewnia niezbędną funkcjonalność oraz możliwość komunikacji szeregowej z pozostałymi modułami, platformą ARM lub komputerem.

Z punktu widzenia użytkownika najistotniejszą częścią całego systemu jest joystick. To on pozwala sterować pojazdem i przetwarza polecenia wydawane przez użytkownika. Jest on zarazem najbardziej skomplikowanym modułem. Schemat blokowy jego budowy przedstawia rysunek 2.1-1. Sercem joysticka jest mikrokontroler Atmel Atmega8, którego podstawowym zadaniem jest odczytanie wartości wychylenia manipulatora oraz przesłanie odpowiednio przetworzonych danych kierunku pojazdu. Przy budowie manipulatora skorzystano z potencjometrów liniowych, które przy

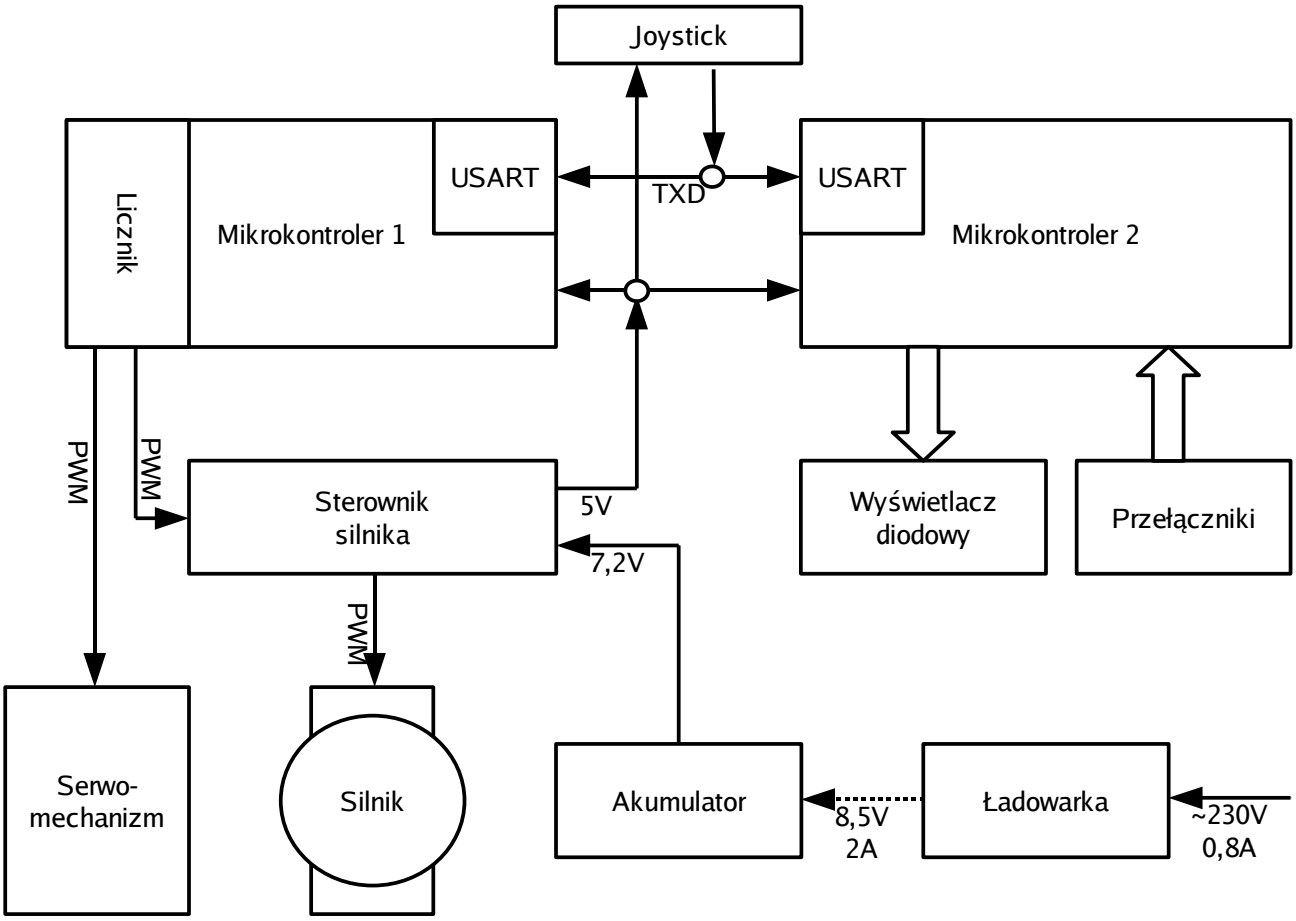
odpowiednim połączeniu tworzą dzielnik napięć. W efekcie takiego podłączenia uzyskujemy liniową zależność: kąt wychylenia – wartość napięcia. Kolejno sygnał przechodzi przez różnicowy wzmacniacz operacyjny, który wzmacnia sygnał do poziomu zapewniającego maksymalną rozdzielczość odczytów. Układ pojedynczego wzmacniacza zawiera dwa potencjometry służące do jak najbardziej optymalnego wysterowania sygnału wychodzącego ze wzmacniacza. Następnie sygnał dociera do przetwornika analogowo cyfrowego, zintegrowanego z mikrokontrolerem, gdzie jego wartość jest przetwarzana na odpowiadającą mu liczbę zapisaną cyfrowo. Odczytane wartości są przeliczane, tak by skrajne położenia drążka zawsze stanowiły tą samą wartość. Tak odczytane i przeliczone wychylenia drążka i przepustnicy, wraz ze stanem przycisków (zwarty, rozwarty) są kolejno wysyłane do układów zainstalowanych na pojeździe.



Rys. 2.1-1. Schemat blokowy joysticka.

Z punktu widzenia pojazdu oraz jego możliwości przemieszczania, najważniejszym modułem staje się układ, który wydaje polecenia bezpośrednio do serwomechanizmu i silnika. Jego schemat blokowy stanowi część schematu znajdującego się na rysunku 2.1-2. Moduł składa się z mikrokontrolera1 oraz sterownika silnika. Jako jednostkę centralną modułu użyto mikrokontrolera Atmel Attiny2313. Mikrokontroler poprzez zintegrowany interfejs komunikacji szeregowej odbiera pakiety danych wysyłane przez joystick oraz na podstawie nich generuje sygnały sterujące. Serwomechanizm wpięty bezpośrednio w układ mikrkonontrolera1, interpretuje otrzymany sygnał ustawiając zadane wychylenie kół. Silnik ze względu na pobieraną moc potrzebuje dodatkowo

układu zwiększającego obciążalność układu. Sterownik silnika jest to gotowy moduł który zawiera w swojej budowie mostek H, wzmacniacze oraz interfejs pozwalający na cyfrowe sterowanie. Do sterownika silnika jest podawany sygnał tego samego typu jak do serwomechanizmu, pozwalając w ten sposób bardzo precyzyjnie ustawiać prędkość obrotową silnika.



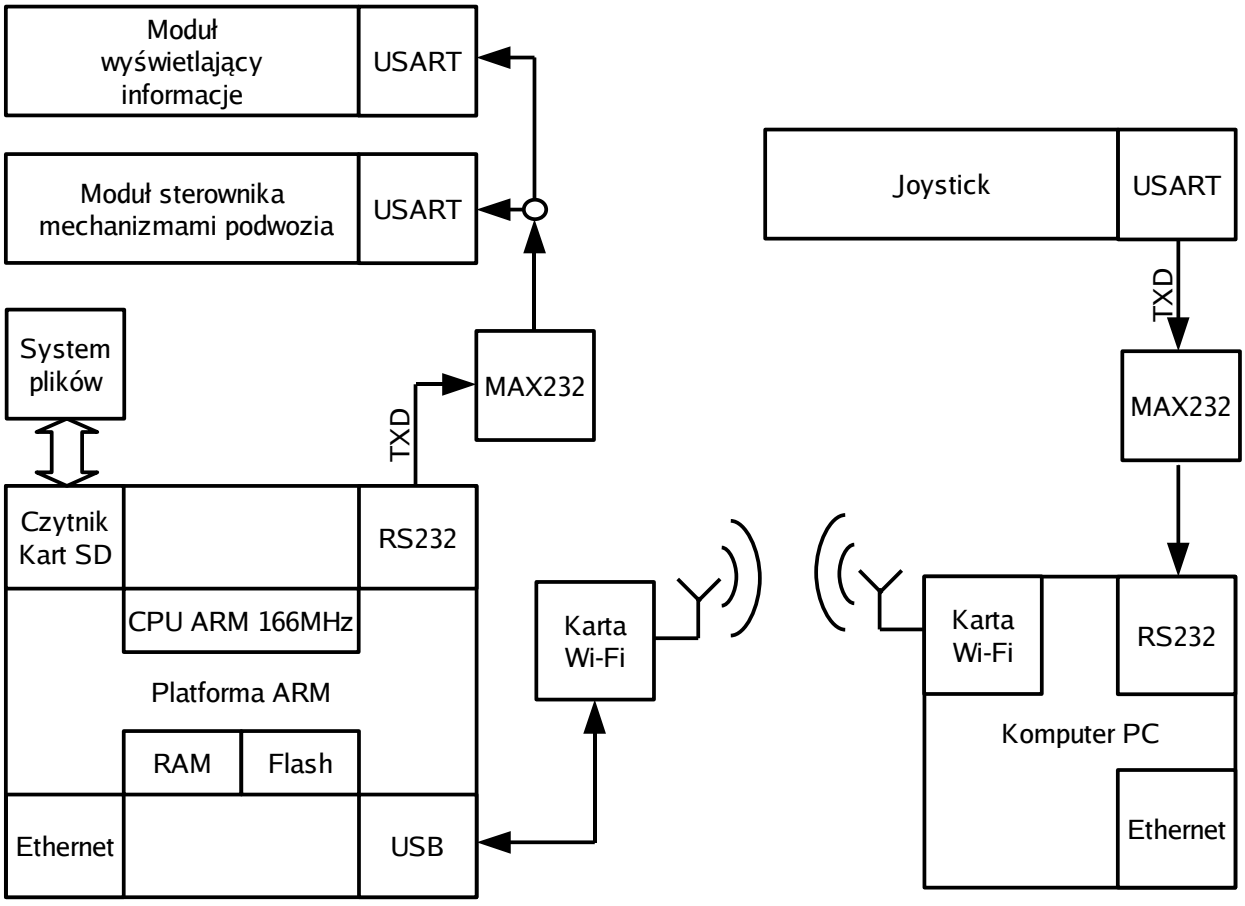
Rys. 2.1-2. Schemat blokowy elektroniki pojazdu.

Ostatnim modułem stanowiącym system sterowania jest układ przedstawiający użytkownikowi dane transmitowane przez joystick. Jego schemat jest również przedstawiony na rysunku 2.1-2 wraz z pozostałymi układami stanowiącymi elektronikę pojazdu. Moduł został zbudowany na bazie mikrokontrolera Atmel Atmega8515, do którego zostały zamontowane przełączniki oraz wyświetlacz diodowy. Moduł ten stanowi kluczową rolę w procesie ustawiania parametrów pracy wzmacniaczy zawartych w joysticku, gdyż w czasie rzeczywistym pokazuje cyfrową interpretację sygnału generowanego przez joystick. Moduł użyteczny jest także podczas wykrywania błędów w pracy w pozostałych modułach.

Całość systemu jest zasilana z wysokoprądowego akumulatora o napięciu 7,2V poprzez sterownik silnika, gdzie napięcie zostaje zmniejszone do poziomu 5V. Wyjątkiem jest silnik który jest zasilany napięciem akumulatora, co pozwala osiągnąć większą moc. Użytkowanie akumulatora

nie wymaga stosowania układów mierzących poziom rozładowania. Akumulator przez cały czas oddaje energię przy napięciu znamionowym, natomiast w momencie rozładowania napięcie na zaciskach drastycznie spada do poziomu bliskiemu 0V. W tym momencie należy naładować akumulatory w celu dalszego użytkowania pojazdu.

Jednym z celów projektu było rozbudowanie systemu o platformę stanowiąca mikrokomputer ARM. Dzięki modułowej konstrukcji oraz wykorzystaniu standardowych interfejsów szeregowych każda część finalnego systemu komunikuje się w taki sam sposób. Jedynie w przypadku elektroniki pojazdu oraz joysticka konieczne jest zastosowanie konwertera napięć MAX232. Całość systemu pracuje w podobny sposób jak we wcześniejszym przypadku, z tą różnicą że dane przepływające w systemie przepływają dodatkowo torem radiowym. Przepływ danych wygląda następująco. Joystick linią szeregową przesyła dane do komputera, które następnie przesyłane są linią radiową do platformy ARM. Mikrokomputer ARM przesyła dane linią szeregową do modułu sterującego.



Rys. 2.1-3. Schemat blokowy finalnego systemu sterowania.

3. Metody sterowania podwoziem

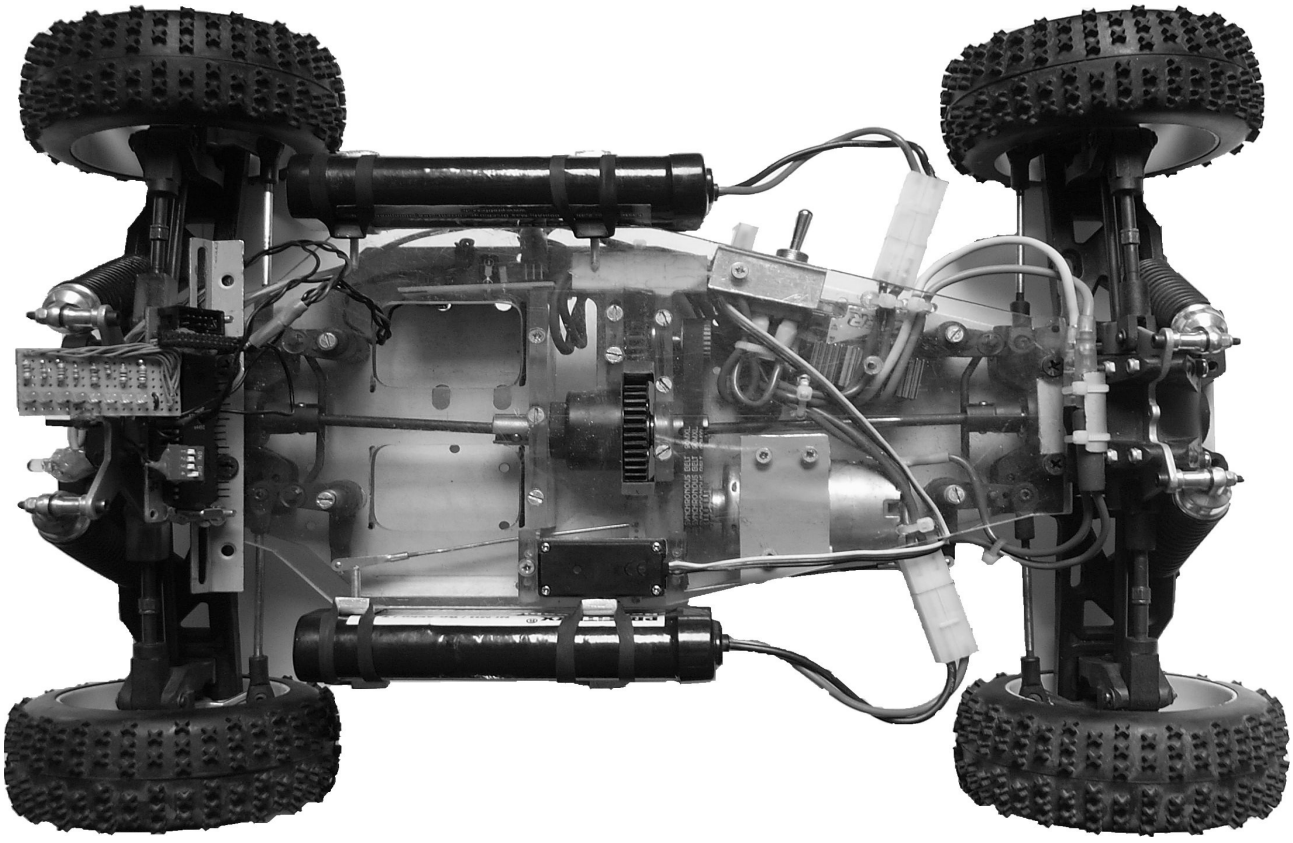
Podwozie pojazdu czterokołowego zostało zbudowane na zamówienie w firmie ModelFan (http://www.modelfan.pl/). Podwozie zostało tak zaprojektowane aby mogło stanowić solidną platformę do dalszej rozbudowy oraz żeby jego parametry jezdne były na wysokim poziomie. Podwozie posiada pełny zakres ruchów, zarówno jazda w przód/w tył jak i skręcanie w obie strony.

Posiadając już mechanicznie gotowe podwozie należało dobrać mechanizm pozwalający ustawić odchylenie kół za pomocą sygnałów elektrycznych. Wybór ten nie był zbyt trudny ponieważ branża modelarska oferowała całą gamę gotowych serwomechanizmów specjalnie stworzonych do tego typu zastosowań. Wspólną cechą wszystkich serwomechanizmów jest ich interfejs sterowania, pozwalający w prosty sposób z bardzo dużą dokładnością ustawiać wychylenie kół. Następnie podwozie należało zaopatrzyć w odpowiedni napęd. Na początku trzeba było się zastanowić nad typem jednostki napędowej. Dostępne są dwie opcje do wyboru, wysokoobrotowy modelarski silnik spalinowy bądź jeden z wielu rodzajów silników elektrycznych. Wybór padł na silnik elektryczny, głównie ze względu na jego prostotę obsługi, bezawaryjność oraz moc dorównującą silnikowi spalinowemu. Zakładano także, że gotowy już pojazd będzie się poruszał głównie w pomieszczeniach zamkniętych, więc hałas i spaliny emitowane z silnika spalinowego byłyby bardzo uciążliwe. Użycie silnika spalinowego sprawiłoby także, że sterowanie pojazdem o takiej mocy w niewielkich pomieszczeniach byłoby wręcz niemożliwe. Po dokonaniu wyboru typu silnika należało znaleźć kompromis pomiędzy mocą i masą całkowitą, a prędkością i przyśpieszeniem pojazdu. Parametry różniły się drastycznie od siebie w zależności od typu silnika (krokowy, liniowy) oraz zastosowanej przekładni.

Następnie należało dobrać metodę sterowania podwoziem, czyli sposób ustawiania odchylenia kół oraz ustawiania prędkości obrotowej silnika, w obie strony. Zadanie to wiązało się z budową i zaprogramowaniem odpowiednich układów elektronicznych. Z jednej strony układu sterującego, tworzącego interfejs dla użytkownika, dzięki któremu użytkownik może w czasie rzeczywistym wpływać na zachowanie się podwozia oraz wszystkich podzespołów na nim zamontowanych. Z drugiej strony układ tłumaczący wszystkie odebrane dane na ruch odpowiednich podzespołów podwozia. Konieczne było także dobranie, lub ewentualne opracowanie, odpowiednich metod kodowania i przesyłania informacji pomiędzy podwoziem a urządzeniem sterującym, zarówno na poziomie sprzętowym jak i programowym.

3.1. Charakterystyka podwozia

Na mechanikę w pojeździe składają się mechanizmy różnicowe, wały napędowe oraz cięgła kierownicze. Wszystko zostało tak zaprojektowane, aby cechowało się dużą trwałością o czym mogą świadczyć na przykład stalowe wały napędowe oraz aluminiowe przekładnie różnicowe. Podwozie jest dość unikatowe ponieważ charakteryzuje się nie tylko tym, iż posiada napęd na cztery koła i niezależne zawieszenie każdego koła, ale także wszystkie koła są skrętne co sprawia, że pojazd posiada wyjątkowo mały promień skrętu, niecały metr (przy długości podwozia ok. 45cm). Przy tak zaprojektowanym układzie kierowniczym, jedynym rozsądnym rozwiązaniem pozwalającym przenieść moc z silnika na koła było zastosowanie trzech dyferencjałów. Dwóch



Rys. 3.1-1. Widok pojazdu z góry.

pomiędzy przednimi i tylnymi półosiami oraz trzeci centralny. Dzięki takiemu rozwiązaniu każde koło pojazdu, przy pokonywaniu nierównego podłoża lub zakrętów, obraca się z różną prędkością, co sprawia że wszystkie przełożenia, wały napędowe i półosie są znacznie mniej obciążone, a także silnik pracuje pod mniejszym obciążeniem. Rozwiązanie to posiada także wady. Największą z nich

jest fakt, że gdy jedno koło straci przyczepność wtedy cały pojazd jest unieruchomiony. Problemowi temu można zaradzić poprzez dobudowanie elektronicznej blokady mechanizmów różnicowych, wtedy cała moc z silnika rozdzielana byłaby równomiernie. Alternatywnym rozwiązaniem byłoby zbudowanie, na każdej pół osi, niezależnego hamulca oraz czujnika prędkości. Rozwiązanie to pozwoliłoby na przyhamowanie koła, które się za szybko obraca względem pozostałych, przekazując tym samym większy moment obrotowy na pozostałe koła.

Rys. 3.1-2. Widok pojazdu z prawej strony.

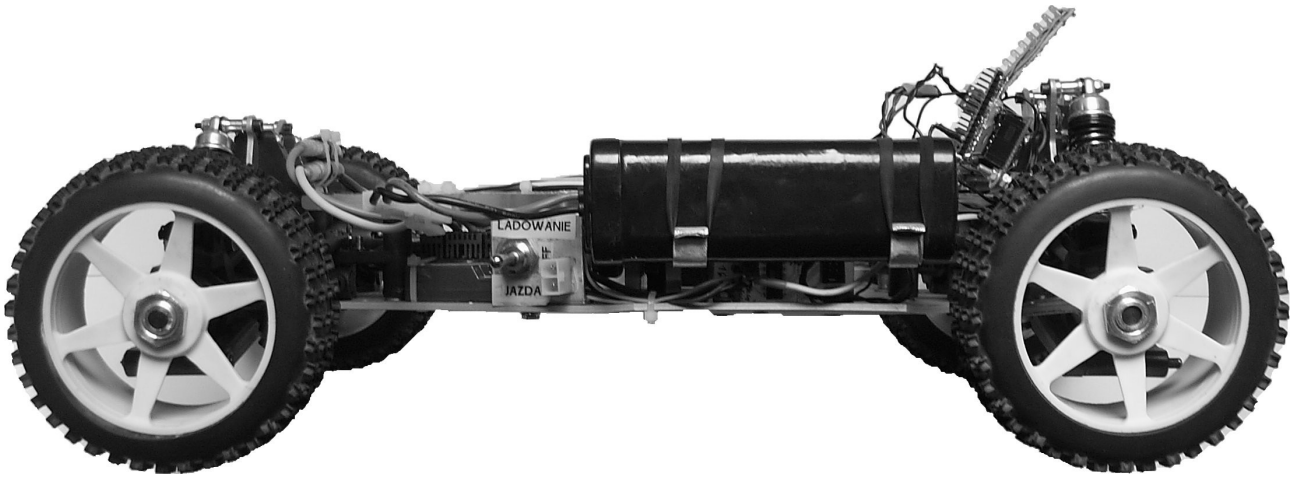
Początkowo jednostkę pojazdu miał stanowić silnik krokowy, rozwiązanie takie uniknęłoby stosowania skomplikowanych w budowie przekładni. Pojazd posiadałby bardzo dużą moc i początkowe przyśpieszenie, kosztem maksymalnej osiąganej prędkości, która w tym wypadku byłaby bardzo mała. Zachowanie takie implikuje charakterystyka pracy silnika krokowego, z której wynika, że im większa jest prędkość obrotowa wału silnika, tym silnik wykazuje mniejszą wartość momentu obrotowego, sprawiając że przy pewnej prędkości moc silnika nie wystarczałaby do przemieszczenia całego pojazdu. Finalnie wybrano elektryczny silnik liniowy prądu stałego firmy JohnsonElectronic (http://www.johnsonelectric.com/) zaprojektowany dla obrabiarek przemysłowych. Wybór tego rodzaju i tego modelu silnika został podyktowany faktem iż jego cena była znacznie niższa niż silników modelarskich, a parametrami dorównywał lub nawet przewyższał niektóre z nich.

Tabela 3.1-1. Krótka charakterystyka silnika PM03028.

Maksymalna prędkość obrotowa	6500rpm
Znamionowe napięcie zasilania	7-14V
Maksymalny pobierany prąd bez obciążenia	~10A
Maksymalny pobierany prąd z obciążeniem	15-20A
Maksymalna wartość momentu obrotowego	160 m Nm

Dokładna charakterystyka zastosowanego silnika zamieszczona jest na stronie producenta (http://www.johnsonelectric.com/product/product_sheet/fect_sheet.php?ProdID=277--A).
Silnik nie jest zbyt duży i ciężki, dlatego idealnie wpasowuje się w podwozie sprawiając, że wolna przestrzeń w podwoziu zostaje optymalnie wykorzystana (Rys. 3.1-1, Rys 3.1-2).

Na układ napędowy dodatkowo składa się szereg przekładni zębatych. Pierwsza z nich jest zarazem najważniejsza. Stanowią ją dwa koła zębate i gumowy pasek zębaty. Na tych właśnie elementach gromadzą się największe siły podczas startu i hamowania. Następnie mamy przekładnię zębatą o dość dużym przełożeniu, która zapewnia znaczną redukcję obrotów, jednocześnie zwiększając moment obrotowy. Moc silnika jest kolejno przekazywana na centralny dyferencjał, przednią i tylną przekładnię różnicową docierając na końcu do kół. Na każdej przekładni różnicowej jest przełożenie ok 1÷5 zapewniając dalszy spadek prędkości obrotowej oraz w efekcie większy moment obrotowy na kołach. Tak zbudowany napęd zapewnia pojazdowi bardzo dobre przyśpieszenie, moc i prędkość maksymalną, która jest na dość dobrym poziomie gdy masa całkowita pojazdu dochodzi do 5kg.

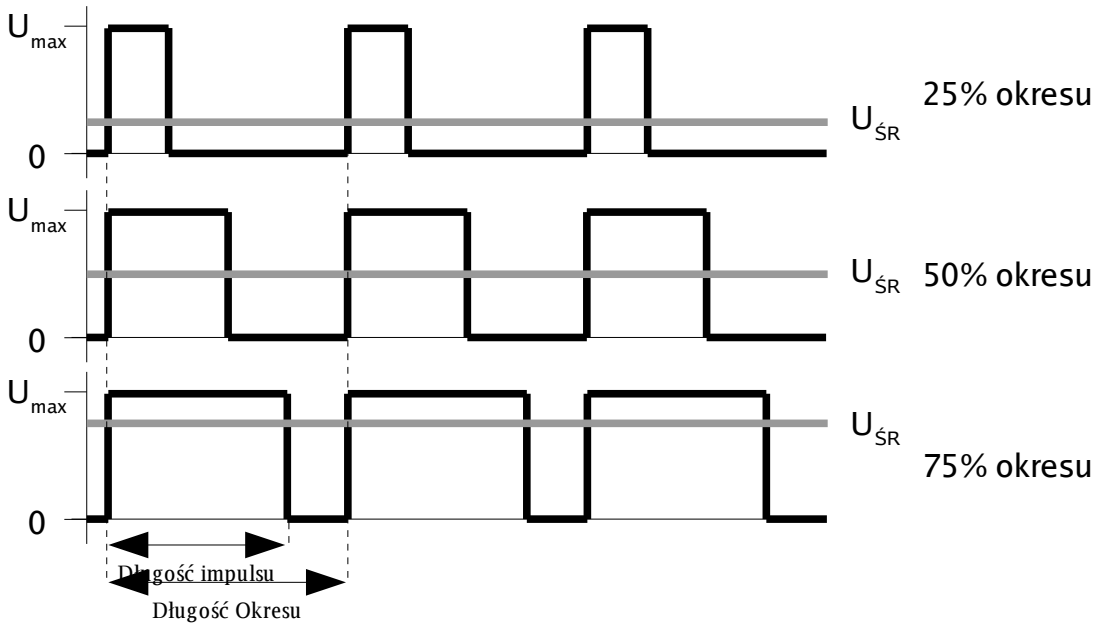


Rys. 3.1-3. Widok pojazdu z lewej strony.

Sercem układu sterowniczego jest serwomechanizm HS-475HB firmy Hitec RCD (http://www.hitecrcd.com/servos/show?name=HS-475HB). Jest to niewielki, niedrogi a zarazem o wystarczających parametrach, serwomechanizm. Jego użycie zostało podyktowane faktem, iż jest to już gotowe bezobsługowe urządzenie, które do pracy potrzebuje jedynie zasilania na poziomie 5V oraz pojedynczego sygnału PWM. Cechy te sprawiają, że zamontowanie i uruchomienie urządzenia nie będzie nastręczać wielu problemów natury mechaniczno-elektronicznej. Serwomechanizm za pomocą szeregu cięgieł z dużą dokładnością ustawia kąt wychylenia kół.

3.2. Dobór metody sterowania

Wybranie metody sterowania serwomechanizmem nie sprawiało większych problemów, ponieważ użyty podzespół narzucał już gotowe rozwiązanie. Podzespół wymagał doprowadzenia zasilania o napięciu 5V oraz jednej linii z sygnałem sterującym. Z informacji umieszczonych na stronie producenta (http://www.hitecrcd.com/product_file/file/21/Servomanual.pdf), można się dowiedzieć, jakiego typu sygnał jest potrzebny oraz jak serwomechanizm reaguje na zmiany

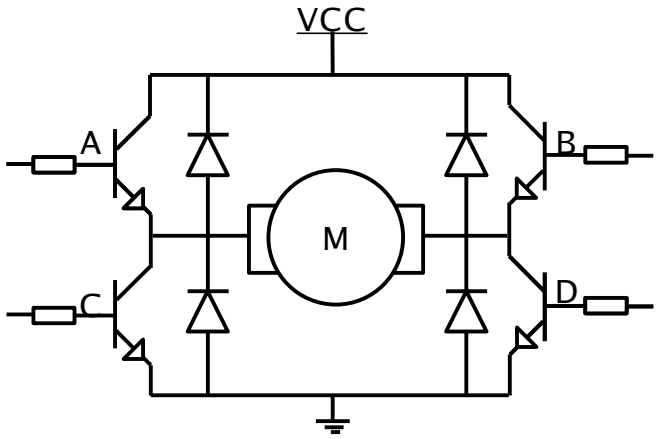


Rys. 3.2-1. PWM - Modulacja szerokości impulsu.

sygnału. Wymaganym sygnałem jest sygnał impulsowo prostokątny, czyli sygnał prostokątny o zmiennej szerokości impulsu. Metoda taka popularnie nazywa się PWM (ang. Pulse-Width Modulation) czyli modulacja szerokości impulsu. Polega ona na regulacji sygnału prądowego lub napięciowego poprzez zmianę szerokości impulsu o stałej amplitudzie. Układ PWM może być połączony z urządzeniem bezpośrednio lub przez filtr dolnoprzepustowy, który wygładza przebieg napięcia lub prądu [2]. Na rysunku 3.2-1 pokazano przykładowe przebiegi napięciowego sygnału PWM, kolorem szarym oznaczono przybliżona amplitudę sygnału przechodzącego przez filtr dolnoprzepustowy. Należało zatem opracować i zbudować układ który będzie w stanie

wygenerować tego typu sygnał oraz będzie możliwa zmiana parametrów generowanego sygnału w czasie rzeczywistym [3].

Zadanie opracowania metody wysterowania pracy silnika okazało się nieco bardziej skomplikowane niż w przypadku serwomechanizmu. Największym problemem okazało się zapewnienie silnikowi odpowiedniej mocy, wiązało się to z takim zaprojektowaniem układu aby posiadał wystarczającą wydajność prądową. Pierwszym pomysłem, a zarazem najbardziej oczywistym było zaprojektowanie układu zasilającego silnik wykorzystującego konstrukcje mostka H. Taka konstrukcja pozwala na możliwość obrotu wału silnika w dowolnym kierunku oraz pracy silnika jako hamulca, co w przypadku pojazdu jest bardzo ważne. Budowa podstawowego układu mostka H jest bardzo prosta, składają się na nią cztery tranzystory kluczujące lub przekaźniki. Często do układu włącza się także diody zabezpieczające układ. Diody maja za zadanie niwelować prądy powstające w cewce silnika podczas odłączania napięcia, które mogłyby przeciążyć cały układ. Przykładowa konstrukcja mostka H znajduje się na rysunku 3.2-2. Zasada działania pokazanego niżej układu jest następująca. W przypadku wyłączenia wszystkich

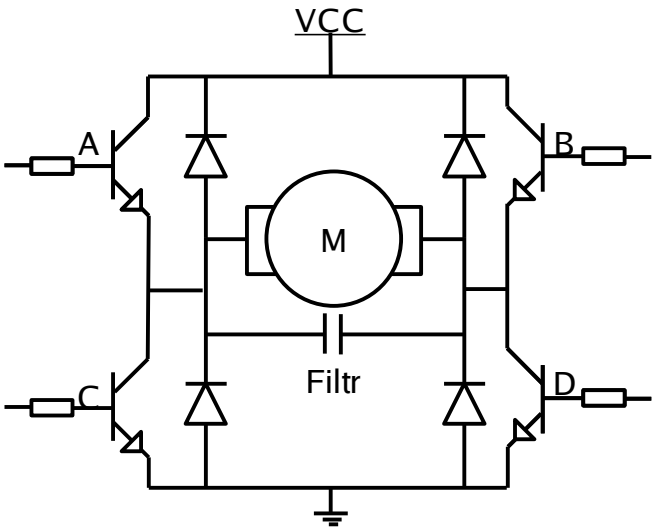


Rys. 3.2-2. Mostek H.

tranzystorów (A, B, C, D) przez układ oraz przez silnik nie przepływa żaden prąd, tym samym wał silnika nie stawia żadnych oporów mechanicznych. W przypadku włączenia tranzystorów A i D prąd przepływa przez cewkę silnika oraz przez te tranzystory, powodując obrót wału silnika. Włączając tranzystory B i C uzyskujemy podobny efekt jak we wcześniejszym przypadku, z tą różnicą, że prąd w cewce silnika przepływa w przeciwną stronę, tym samym wał obraca się w stronę przeciwną. W przypadku gdy włączymy tranzystory A i B, przez cewkę silnika nie będzie przechodził prąd, powodując jednak pojawienie się znacznych oporów podczas obrotu wału. Przypadek ten można wykorzystać jako hamulec elektryczny. Zabronione jest jednoczesne

włączanie tranzystorów A i C lub B i D nastąpiłoby wtedy zwarcie, przyczyniając się do przepływu przez układ prądu o znacznej wartości, a w rezultacie uszkodzenie tranzystorów oraz prawdopodobne uszkodzenie źródła zasilania. [3]

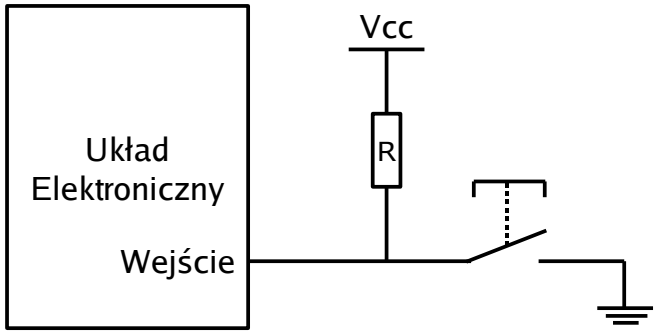
Samo wykorzystanie mostka H nie jest wystarczające, ponieważ układ tego typu daje jedynie możliwość pracy silnika w czterech stanach: wirowanie w prawo, wirowanie w lewo, spoczynek oraz hamowanie. Konieczne było rozbudowanie układu zasilającego silnik o możliwość ustawiania prędkości obrotowej silnika w dowolnym zakresie ilości obrotów na minutę, od zera do maksymalnej wartości. Pierwszym rozwiązaniem była modyfikacja układu mostka H w taki sposób, że zastosowane tranzystory pracują w pełnym zakresie przewodzenia. Rozwiązanie takie pozwoliłoby sterować prędkością obrotową silnika poprzez zmianę wartości napięcia dostarczanego na styki silnika. Jednak już na etapie obliczeń okazało się, że przy małych prędkościach moc wytracana na tranzystorach byłaby bardzo duża, powodując przegrzewanie się tranzystorów



Rys. 3.2-3. Mostek H rozbudowany o filtr dolnoprzepustowy.

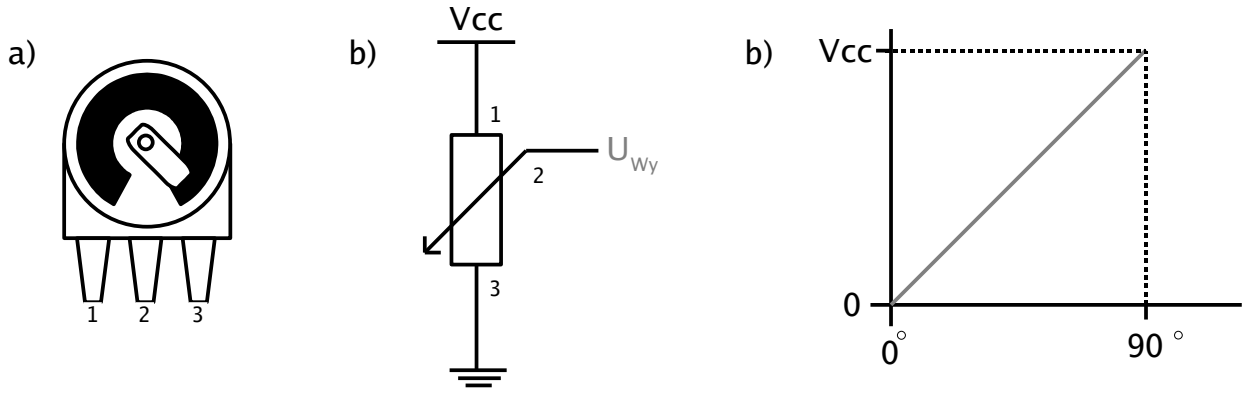
i w efekcie ich uszkodzenie. Kolejnym rozwiązaniem było użycie układu mostka H, sterowanego sygnałem typu PWM oraz rozbudowanego o filtr dolnoprzepustowy wpięty do układu równolegle do cewki silnikia. Rozbudowany układ znajduje się na rysunku 3.2-3. Sterowanie układem odbywa się w podobny sposób jak we wcześniejszym rozwiązaniu, z tą różnicą, że zamiast włączać tranzystory stałym napięciem, włączamy je sygnałem PWM. W uproszczeniu Mostek H działa w tym przypadku jak wzmacniacz napięciowy. Zastosowany filtr wygładza przebieg napięcia, które zachowuje się tak jak napięcie oznaczone kolorem szarym na rysunku 3.2-1. Silnik będąc zasilany napięciem ciągłym generuje mniej zakłóceń elektromagnetycznych, które mogłyby spowodować uszkodzenie układów elektronicznych oraz diametralnie zmniejszyć żywotność samego silnika.

Znając już metody sterowania pracą podzespołów pojazdu należało się zastanowić nad urządzeniem, które pozwalałoby dokładnie wysterować ruchy podwozia. Urządzenie powinno umożliwiać pobranie informacji od użytkownika, przetwarzanie ich oraz wysłanie odpowiednich sygnałów sterujących. Jako pierwsze nasuwało się rozwiązanie wykorzystujące joystick cyfrowy, lub klawiaturę komputera. Rozwiązania te cechują się bardzo dużą prostotą budowy i obsługi gdyż w obu przypadkach polegają jedynie na odczytaniu stanu mikrostyku, znajdującego się pod przyciskiem klawiatury lub pod drążkiem joysticka. Mikrostyk może znajdować się w danym czasie tylko w jednym z dwóch stanów, zwarty bądź rozwarty. Na rysunku 3.2-4 przedstawiono układ pozwalający na odczyt stanu mikrostyku. W pozycji rozwartej potencjał na wejściu układu jest równy napięciu zasilania, odczytujemy stan wysoki czyli logiczna jedynkę. W przypadku kiedy mikrostyk znajduje się w pozycji zwartej (wciśnięty przycisk na klawiaturze, przechylony drążek) potencjał na wejściu ustala się na poziomie potencjału masy układu. Odczytujemy stan niski czyli logiczne zero. Przedstawiony typ rozwiązania zagadnienia konstrukcji sterownika okazuje się nie być wystarczająco dokładny. Na wejściu układu pojawiają się jedynie dwa stany, wysoki i niski, mówiące o włączeniu lub wyłączeniu układu. Stosując przedstawione rozwiązanie sterowanie pojazdem stałoby się bardzo ograniczone, jedynie do kilku podstawowych stanów. Na przykład ustawienie stopnia wychylenia kół ograniczałoby się jedynie do trzech pozycji, skrajnej lewej, skrajnej prawej oraz środkowej.



Rys. 3.2-4. Układ odczytu stanu mikrostyku.

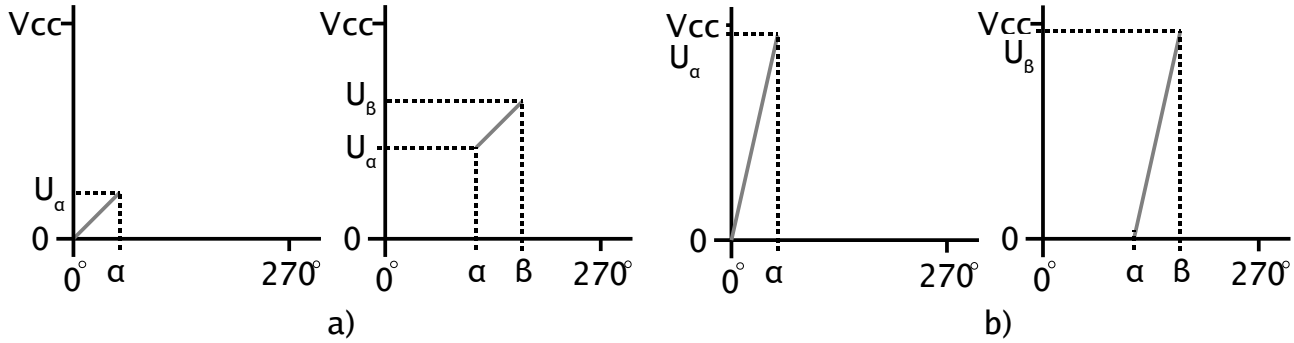
Zważywszy na ograniczenia sterowania cyfrowego, postanowiono przyjrzeć się sterowaniu wykorzystującemu układy analogowe. Metoda sterowania analogowego jest dużo bardziej dokładna, uzyskuje się dzięki niej wszystkie stany pośrednie pomiędzy wartościami skrajnymi. Jednak konstrukcja analogowych sterowników często jest dużo bardziej skomplikowana niż rozwiązań cyfrowych. Sterowniki analogowe to przede wszystkim aparatury do sterowania modeli radiowych oraz różnego rodzaju joysticki analogowe i wolanty. Wspólną cechą wszystkich przedstawionych sterowników analogowych jest ich typ budowy wewnętrznej, a dokładniej układ



Rys. 3.2-5. a) Schematyczna budowa potencjometru jedno-obrotowego (kąt obrotu 270˚). b) Układ zastosowania potencjometru jako dzielnika napięcia. c) Wykres napięcia wyjściowego jako funkcji wychylenia.

przetwarzający wartość kątową wychylenia drążka na sygnał elektryczny. Głównym elementem najprostszego układu jest potencjometr liniowy jednoobrotowy, który zmienia wartość swojej rezystancji wprost proporcjonalnie do kąta obrotu pokrętła potencjometru. Do tego typu układów najczęściej wykorzystywanymi potencjometrami są potencjometry o zakresie obrotu niecałych 90˚, które są bezpośrednio przymocowane do drążka w punkcie osi obrotu. Zasada działania potencjometru wygląda następująco. Do pokrętła przymocowany jest styk, który w momencie obracania pokrętłem wędruje po ścieżce oporowej, zmieniając rezystancje pomiędzy wyprowadzeniami ścieżki oporowej (skrajne wyprowadzenia), a wyprowadzeniem styku (środkowe wyprowadzenie). Uproszczona budowa potencjometru została przedstawiona na rysunku 3.2-5a). W podejściu praktycznym najczęściej potencjometr jest wpinany do obwodu tak by na jego wyprowadzeniach kształtował się dzielnik napięcia. Tym samym zmienia się funkcja wyjściowa potencjometru z funkcji rezystancji na funkcje napięcia zależną od kąta obrotu. Na rysunku 3.2-5b) przedstawiono układ potencjometru pracującego jako dzielnik napięcia, który kształtuje wartość napięcia wyjściowego w funkcji rezystancji. Funkcja napięcia wyjściowego jest funkcją opisaną wzorem $U_{WY} = \frac{R_{23}}{R_{12} + R_{23}} * V_{CC}$, ($R_{xy}$ – rezystancja pomiędzy stykami x i y). Znając liniową zależność napięcia i rezystancji oraz zależność rezystancji do kąta obrotu, można wygenerować wykres funkcji napięcia zależnej od kąta obrotu, przedstawiony na rysunku 3.2-5c).

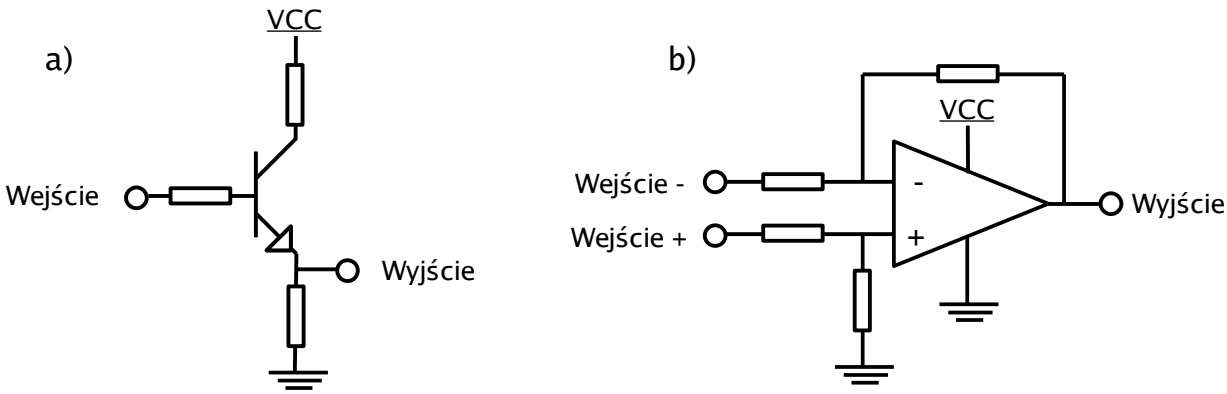
Następnym elementem układów elektronicznych joysticka może być wzmacniacz, który jest zbędny w przypadku gdy pokrętło potencjometru wykonuje ruch w swoim pełnym zakresie obrotów. W przypadkach gdy potencjometr posiada większy kąt obrotu np. 270˚, pokrętło wykonuje niewielki obrót, wykorzystana zostaje tylko część zakresu obrotu. Zatem zakres napięcia wyjściowego zostaje znacznie zmniejszony (Rys. 3.2-6a).).



Rys. 3.2-6. Wykres napięcia jako funkcji wychylenia.
a) Przebieg na wejściu wzmacniacza.
b) Przebieg na wyjściu wzmacniacza.

W tego typu przypadkach, dla uzyskania większej dokładności sterownika, układ wyposaża się we wzmacniacz. Wzmacniacz jest układem w pełni analogowym, którego zadaniem jest wytworzenie na wyjściu układu sygnału o wartości proporcjonalnie większej do sygnału wejściowego. Do wejścia wzmacniacza doprowadza się sygnał z dzielnika napięć, natomiast na wyjściu otrzymujemy sygnał o proporcjonalnie większej wartości. Wartość otrzymanego sygnału zmienia się od poziomu bliskiego 0V, do poziomu nieznacznie mniejszego od poziomu napięcia zasilania. Konstrukcja najprostszego wzmacniacza oparta jest na pojedynczym odpowiednio wysterowanym tranzystorze. Na rysunku 3.2-7a). został przedstawiony przykładowy układ wzmacniacza opartego na tranzystorze bipolarnym w układzie wspólnego kolektora. Zasada działania tranzystora bipolarnego polega na sterowaniu prądem kolektora przy pomocy prądu bazy, wiec im większy prąd jest podany na bazę tym większy prąd wpłynie przez kolektor do tranzystora. Z praktycznego punktu widzenia można przyjąć stwierdzenie, że tranzystor w przedstawionym układzie tworzy sterowany prądem dzielnik napięcia. Stwierdzenie takie jest prawdziwe ponieważ, układ jest zawsze zasilany napięciem o niezmiennej wartości. Natomiast z prawa Ohma wynika zależność, że gdy napięcie jest stałe, rezystancja tranzystora jest odwrotnie proporcjonalna do prądu przez niego przepływającego. Dlatego im większy prąd bazy, tym większy prąd kolektora, jednocześnie zmniejsza się rezystancja tranzystora oraz zwiększa się napięcie na wyjściu układu [6].

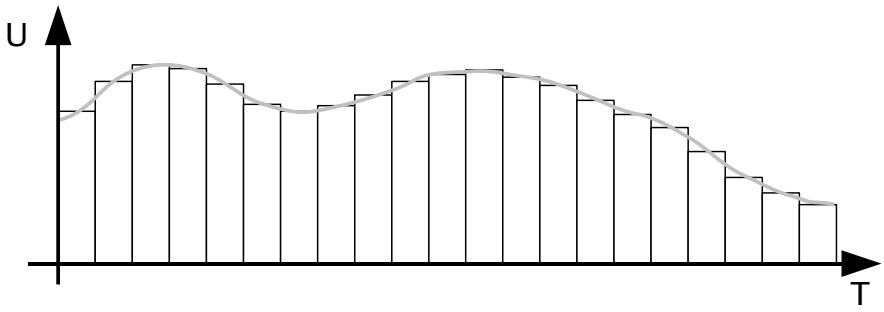
Kolejnym rozwiązaniem budowy wzmacniacza, jest użycie mikro układu scalonego, który zawiera w swojej budowie wzmacniacz operacyjny. Wzmacniacz operacyjny to wielostopniowy, różnicowy wzmacniacz prądu stałego, charakteryzujący się bardzo dużym różnicowym wzmocnieniem napięciowym i jest zwykle przeznaczony do pracy z zewnętrznym obwodem sprzężenia zwrotnego, który decyduje o głównych właściwościach całego układu (def. z [2]). Układ wzmacniacza



Rys. 3.2-7. Przykładowe konstrukcja wzmacniacza.
a) Wzmacniacz tranzystorowy w układzie OC.
b) Wzmacniacz operacyjny w układzie różnicowym.

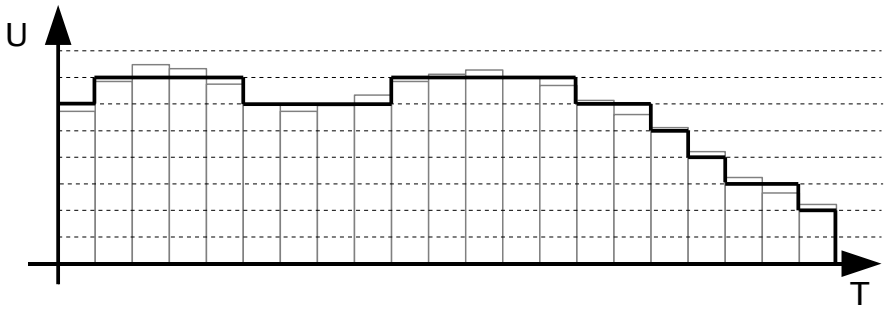
operacyjnego przedstawiony jest na rysunku 3.2-7b). Układ posiada dwa wejścia, nazwane odpowiednio nieodwracające („+") oraz odwracające („-"), daje to możliwość połączenia dwóch źródeł sygnału. We wzmacniaczu operacyjnym wzmocnieniu podlega jedynie różnica sygnałów podanych na oba wejścia, wspólna składowa sygnałów jest przytłumiana do wartości bliskiej 0V [4]. Wzmocnienie całego układu jest regulowane poprzez rezystor sprzęgający wyjście wzmacniacza z wejściem nieodwracającym. Sprawiając ze uruchomienie całego układu staje się sprawą bardzo łatwą. Zastosowanie praktyczne wzmacniaczy operacyjnych niesie ze sobą szereg korzyści takich jak, łatwość konfiguracji układu, większa liniowość pracy niż w przypadku tranzystora. Największa zaletą wzmacniaczy operacyjnych jest możliwość prostego zbudowania, układu precyzyjnego wzmacniacza różnicowego. Dzięki zastosowaniu wzmacniacza sygnału, małe zmiany pozycji pokrętła potencjometru generują duże zmiany napięcia na wyjściu układu (Rys. 3.2-6b).), uzyskując tym samym znacznie większą dokładność sterownika.

Następnym krokiem przy budowaniu analogowego sterownika, jest zamiana sygnału analogowego na cyfrowy. Czynność ta jest konieczna ze względu na fakt, iż zero-jedynkowe układy elektroniczne nie są w stanie wykonywać jakichkolwiek operacji na sygnałach analogowych. Proces przetworzenia sygnału analogowego na sygnał cyfrowy dzieli się na dwa etapy. Pierwszym etapem jest próbkowanie, czyli proces stworzenia sygnału impulsowego, reprezentującego sygnał ciągły [2]. Próbkowanie, inaczej kwantowanie w czasie, polega na zmianie sygnału zmieniającego się w sposób ciągły, na sygnał dyskretny, zmieniający się w sposób skokowy. W praktyce układ próbkujący w ustalonych odstępach czasu mierzy wartość chwilową sygnału, i na jej podstawie, tworzone są próbki. Wszystkie próbki składają się na przebieg sygnału dyskretnego. Na rysunku



Rys. 3.2-8. Przykład próbkowania sygnału analogowego.

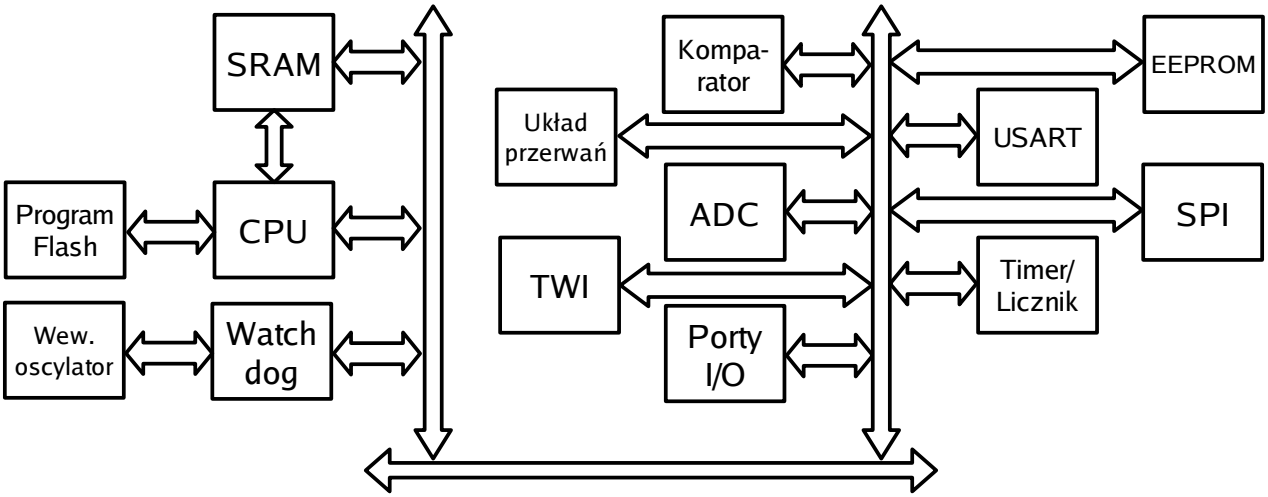
3.2-8. przedstawiono przykład próbkowania sygnału ciągłego (kolor szary). Przetworzenie sygnału na dyskretny sprawia, że nieodwracalnie jest tracona cześć informacji zawartych w pierwotnym sygnale. Drugi etap przetwarzania informacji analogowej na postać cyfrową stanowi kwantyzacja. Polega ona na ograniczeniu zbioru wartości przyjmowanych przez sygnał analogowy. Ograniczenie to wynika z faktu, iż wartość chwilowa sygnału może przyjąć dowolną wartość, natomiast systemy cyfrowe potrafią jedynie operować na zbiorach o skończonej liczbie elementów. W przypadku systemów cyfrowych elementem zbioru jest jedna z możliwych do przyjęcia wartości sygnału, zwanymi poziomami kwantowania. Proces kwantowania ma za zadanie przypisanie wartościom sygnału analogowego, najbliższych poziomów kwantowania. Podczas tego procesu każdemu poziomowi przypisywana jest określona liczba. Kwantyzacja sygnału dyskretnego powoduje dalszą nieodwracalną utratę części informacji. Na rysunku 3.2-9. przedstawiono proces kwantyzacji sygnału dyskretnego z poprzedniego przykładu. Porównując przebieg sygnału analogowego (Rys. 3.2-8. kolor szary) z przebiegiem otrzymanego sygnału cyfrowego (Rys. 3.2-9. kolor czarny) widać, że sygnał cyfrowy różni się w znacznym stopniu od sygnału analogowego. Największy wpływ na wielkość tej różnicy ma proces kwantyzacji. Im większa jest ilość poziomów kwantyzacji tym różnica jest mniejsza [2].



Rys. 3.2-9. Przykład kwantyzacji sygnału dyskretnego.

W przetwornikach analogowo cyfrowych ilość poziomów kwantyzacji określa rozdzielczość przetwornika, która jest podawana w ilości bitów. Na przykład jeśli przetwornik posiada rozdzielczość ośmiu bitów to znaczy, że sygnał jest kwantowany na 256 (2^8) poziomach. Kolejnym czynnikiem wpływającym na wielkość różnicy między przebiegani sygnałów, jest częstotliwość próbkowania, czyli czas pomiędzy kolejnymi odczytami wartości chwilowej sygnału. Im częstotliwość jest większa, czyli odstęp czasu jest mniejszy, tym większe podobieństwo przetworzonego sygnału do sygnału pierwotnego. [2]

Aby system sterowania był kompletny należy go wyposażyć w układ mikroprocesorowy, który będzie kontrolował wszystkie procesy, zarówno pobierania danych z układów analogowych, jak i wysyłania odpowiednich sygnałów do układów wyjściowych. W chwili obecnej na rynku jest bardzo dużo różnego typu układów mikroprocesorowych. Począwszy od układów scalonych małej skali integracji, które zawierają tylko mikroprocesor ośmiobitowy, kończąc na układach bardzo dużej skali integracji, gdzie w jednej obudowie zintegrowano wieloprocesorowy system 32-bitowy wraz z różnego rodzaju układami peryferyjnymi. Mając na uwadze niedużą komplikacje przyszłego systemu sterowania postanowiono zawęzić poszukiwanie układu mikroprocesorowego, do grupy mikrokontrolerów ośmiobitowych. Mikrokontroler jest to komputer wykonany w jednym układzie scalonym, używany do sterowania urządzeniami elektronicznymi. Układ oprócz jednostki centralnej CPU posiada zintegrowaną różnego rodzaju pamięć oraz układy wejścia-wyjścia. Mikrokontroler stanowi użyteczny i całkowicie autonomiczny system mikroprocesorowy, który z reguły do swej pracy nie potrzebuje dodatkowych układów. Może bezpośrednio współpracować z różnymi urządzeniami zewnętrznymi. Wśród wbudowanych w mikrokontroler bloków funkcjonalnych można znaleźć: jednostkę obliczeniową, pamięć danych i programu, liczniki, kontrolery przerwań, kontrolery transmisji szeregowej lub równoległej, przetworniki analogowo-cyfrowe lub cyfrowo-analogowe,, itp [2]. Podstawowe komputery jednoukładowe dostępne na rynku pomimo wielu cech wspólnych różnią się zintegrowanymi peryferiami oraz mogą być konfigurowane w różny sposób. Przykładem takich różnic może być źródło sygnału zegarowego procesora. Sygnał taktujący procesor mikrokontrolera może pochodzić z wewnątrz lub zewnątrz układu. Większość dostępnych układów posiada możliwość pracy na wewnętrznym generatorze częstotliwości, który pozwala na prace układu z częstotliwością do kilkudziesięciu MHz. Jednak przy niektórych zastosowaniach lepsze efekty daje użycie zewnętrznego generatora częstotliwości. Czasem zintegrowany zegar okazuje się nie być wystarczająco dokładny, gdyż jego praca jest zależna od takich czynników zewnętrznych jak np. temperatura. Pozaukładowe taktowanie procesora może być zrealizowane przy pomocy, układu rezonatora kwarcowego z dwoma kondensatorami bądź poprzez inny generator przebiegu prostokątnego [7][10][11].



Rys. 3.2-10. Przykład budowy wewnętrznej komputera jednoukładowego.

Program wykonywany przez mikrokontroler jest zapisany w dostępnej dla układu pamięci. Zastosowana pamięć układu może być pamięcią tylko do odczytu (ROM), pamięcią jednokrotnego zapisu (PROM) lub pamięcią wielokrotnego zapisu (EPROM, EEPROM, FLASH). Proces zaprogramowania pamięci programu może się odbywać na etapie produkcji mikrokontrolera, bądź przez użytkownika w dowolnym momencie w zależności od zastosowanej technologii. Jedną z możliwości programowania mikrokontrolerów jest metoda zapisu pamięci programu bez wyjmowania mikrokontrolera z układu. Metoda ta w zależności od producenta układu jest nazywana ISP (In System Programming) lub ICSP (In-Circut Serial Programing). Mimo faktu iż bardzo duża ilość firm produkuje układy, zdecydowana większość układów jest z sobą zgodna, gdyż zachowane są standardy wprowadzone przez firmę Intel, która jako pierwsza wprowadziła na rynek mikrokontroler jednoukladowy [2].

3.3. Charakterystyka zastosowanej metody

Podczas procesu projektowania systemu sterowania jedną z pierwszych decyzji jaką trzeba było podjąć, był wybór mikrokontrolera, gdyż w głównej mierze od charakterystyki mikrokontrolera zależy budowa i funkcjonowanie przyszłego systemu. Na charakterystykę mikrokontrolera składa się możliwa moc obliczeniowa, ilość i typ zintegrowanych w mikrokontrolerze układów peryferyjnych, ilość zintegrowanej pamięci RAM, pamięci programu oraz pamięci danych. W przypadku budowy systemu sterowania tego typu, prędkość procesora staje się parametrem drugorzędnym, ponieważ nie będzie konieczności wykonywania przez system

Tabela 3.3-1. Cechy architektury RISC [2].

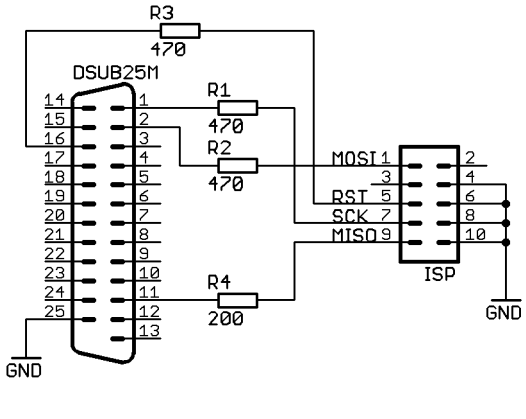
- Zredukowana liczba rozkazów (RISC – kilkadziesiąt, CISC – ok. setki).
- Zredukowane tryby adresowania.
- Ograniczenie komunikacji pomiędzy pamięcią a procesorem. Dane są przesyłane tylko pomiędzy rejestrami a pamięcią.
- Zwiększenie liczby rejestrów do ilości 32,196 i więcej.
- Znacznie skrócony czas wykonywania rozkazów, większość rozkazów wykonuje się w jednym cyklu zegarowym.

skomplikowanych i czasochłonnych obliczeń. Jednocześnie ilość pamięci o swobodnym dostępie także niemiała większego znaczenia, gdyż wszystkie przeglądane mikrokontrolery posiadały ilość pamięci kilka razy większa niż uruchomiony program by wykorzystywał. W takim przypadku jednym z najważniejszych parametrów opisujących mikrokontroler stała się ilość różnych układów peryferyjnych wchodzących w skład układu. Optymalnym rozwiązaniem byłoby użycie układu zawierającego wszystkie potrzebne peryferia takie jak licznik, układ czasowy, przetwornik analogowo cyfrowy, interfejs komunikacji szeregowej oraz ilość portów ogólnego użytku. Przy wyborze układów postanowiono wybór ograniczyć do rodziny mikrokontrolerów AVR firmy ATMEL (http://www.atmel.com/products/avr/). ATMEL AVR to rodzina ośmiobitowych mikrokontrolerów opartych na schemacie procesora RISC, z uwzględnieniem cech architektury harwardzkiej. W tabeli 3.3-1 pokrótce przedstawiono główne cechy architektury RISC. Właściwości te bezpośrednio przekładają się na znacznie szybszą i wydajniejszą prace procesora. Dzięki zmniejszonej liczbie rozkazów oraz zredukowanym trybom adresowania, osiągnięto znacznie uproszczony dekoder rozkazów, jednocześnie kody rozkazów stały się prostsze. Wprowadzony został także tryb adresowania, który znacznie ogranicza ilość przesłań. Większość operacji wykonywana jest według następującego schematu $rejestr_C$=$rejestr_A$ operacja $rejestr_B$, natomiast w architekturze CISC domyślnym operandem jest argument docelowy co wymusza dodatkowo użycie dodatkowego rozkazu przesuwającego dane. Wprowadzone ograniczenie komunikacji pomiędzy pamięcią a procesorem, sprowadza się do przesyłania danych z pamięci do rejestrów procesora, i odwrotnie. Zdefiniowane zostały oddzielne rozkazy do obsługi pamięci (zapis, odczyt), natomiast wszystkie pozostałe rozkazy mogą operować tylko na rejestrach.

Jednak dzięki znacznie zwiększonej ilości rejestrów procesora ilość odwołań do pamięci jest znacznie mniejsza. Możliwość wykonywania rozkazów w jednym takcie zegara, umożliwiło zastosowane przetwarzanie potokowe, które także znacznie uprościło blok wykonawczy. [2]

Rodzina mikrokontrolerów AVR jest bardzo duża, poszczególne jej modele różnią ilością pamięci RAM,EEPROM, FLASH oraz przede wszystkim wyposażeniem w urządzenia peryferyjne. Rodzinę tworzy kilka grup układów. TINY są to najmniejsze układy, które z faktu posiadania najmniejszej ilości peryferii oraz pamięci, wykorzystywane są w niewielkich układach. MEGA jest to grupa mikrokontrolerów charakteryzujących się znaczną rozbudową o liczne urządzenia peryferyjne oraz bardzo duże ilości pamięci. Ostatnią grupę stanowią mikrokontrolery specjalne, które są wyposażone np. w sprzętowy interfejs USB, sterownik wyświetlaczy LCD, dokładne przetworniki ADC, bądź kilkukanałowe generatory PWM [2][10].

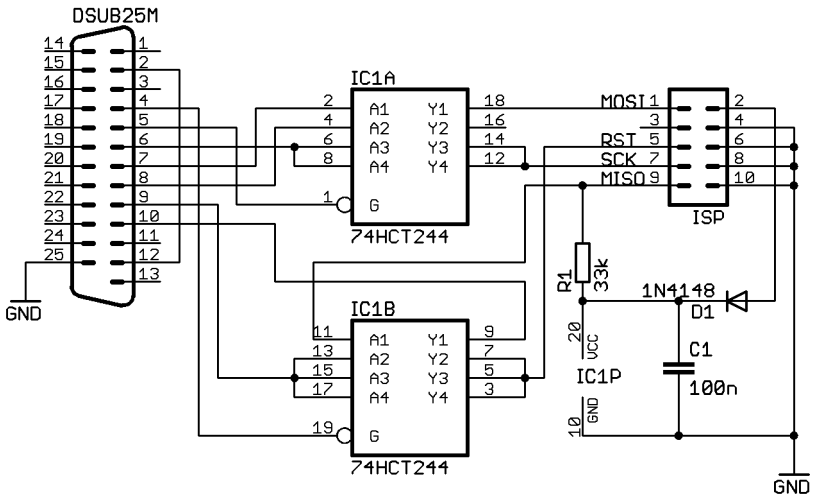
Na wybór mikrokontrolerów rodziny AVR wpłynęły także takie cechy jak, darmowe zaawansowane środowiska programistyczne, darmowe ogólnodostępne programatory, łatwo dostępna szeroka wiedza z zakresu programowania i uruchamiania układów AVR. Producent układów, firma ATMEL udostępnia na swojej stronie internetowej zintegrowane, kompleksowe środowisko programistyczno-uruchomieniowe z możliwością debuggingu, o nazwie AVR studio (http://atmel.com/dyn/products/tools_card.asp?tool_id=2725). Jednak zdecydowano się nie korzystać z firmowego środowiska AVR, gdyż jest ono uruchamiane jedynie na platformie Windows oraz wymaga zakupienia kosztownego programatora. Do kompilowania programów użyto darmowej przystosowanej, do pracy z układami AVR, wersji kompilatora GNU GCC (http://gcc.gnu.org/) dostępnej pod nazwą AVR GCC. Dodatkowo należało budowane środowisko programistyczne zaopatrzyć w biblioteki języka C, AVRlibc (http://www.nongnu.org/avr-libc/) która jest w pełni darmową biblioteką pozwalającą wykorzystać wszystkie możliwości



Rys. 3.3-1. Najprostszy programator układów AVR zgodny z STK200.

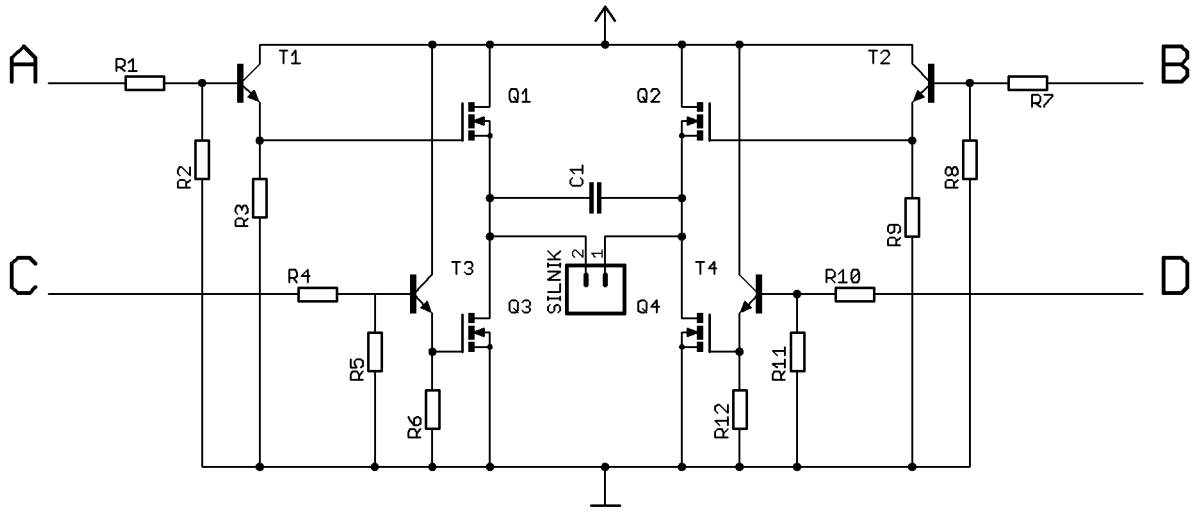
mikrokontrolerów ATMEL AVR. Ostatnim elementem wymaganym do poprawnej pracy środowiska jest pakiet programów Binutils (http://www.gnu.org/software/binutils/). Dokładny opis skompilowania i uruchomienia środowiska programistycznego znajduje się na polskiej stronie AVR-GCC (http://avr.elektroda.eu/?q=node/24) [8][9].

Firma ATMEL w swoich układach z rodziny AVR udostępniła opcje programowania mikrokontrolera w układzie (ISP – In system programing). Rozwiązanie to umożliwia zaprogramowanie mikrokontrolera bez konieczności demontowania go z układu w którym ma działać. ISP znacznie ułatwia zmianę oprogramowania mikrokontrolera, jednocześnie sprawiając, że nie są wymagane specjalistyczne układy programujące. Najpopularniejsze układy programujące to układy STK200 oraz zgodne z nimi. Najprostszy układ programujący składa się jedynie z czterech rezystorów (Rys. 3.3-1.), jednak układ ten jest mało dokładny i podczas procesu programowania mogą występować błędy, czasem uniemożliwiając poprawne zaprogramowanie układu. Kolejny przedstawiony układ (Rys. 3.3-2.) został rozbudowany o trójstanowe bufory, które w znacznym stopniu zapobiegają występującym wcześniej błędom. Przedstawione układy programujące obsługiwane są przez większość ogólnie dostępnych programatorów takich jak np. PonyProg (http://www.lancos.com/prog.html), AVRdude (http://savannah.nongnu.org/projects/avrdude/) bądź UISP (http://www.nongnu.org/uisp/).



Rys. 3.3-2. programator STK200 dla układów AVR.

Decydując się na metodę sterowania silnikiem przy użyciu mostka H, sterowanego sygnałem PWM, należało znacznie przebudować konstrukcję mostka przedstawioną na rysunku 3.2-3. Przebudowa jest konieczna ze względu na fakt iż wykonany układ, według podanego schematu, bardzo szybko uległ uszkodzeniu. Uszkodzenie było spowodowane przepływem przez tranzystory



Rys. 3.3-3. Schemat mostka H wykorzystującego tranzystory MOSFET.

prądów rzędu kilkudziesięciu amper oraz w drugiej kolejności niedokładnym dobraniem punktu pracy tranzystora. Biorąc pod uwagę duży pobór prądu przez silnik oraz fakt, że zwykłe tranzystory bipolarne nie zapewniłyby dostatecznie dużej mocy. Konieczne było zastanowienie się nad wykorzystaniem tranzystorów unipolarnych dużej mocy, które są w stanie przenosić tak wielkie obciążenia prądowe. Jednak takowe rozwiązanie rodzi nowe problemy, gdyż tranzystory tego typu wymagają wysokiego napięcia, dostarczanego na bramę tranzystora, aby przejść w stan aktywny. Konieczne w tym wypadku staje się użycie dodatkowych tranzystorów małej mocy, które odpowiednio włączałyby wysokoprądowe MOSFET'y. Użycie dodatkowych tranzystorów jest podyktowane charakterystyką zastosowanych tranzystorów mocy, ponieważ sygnał z mikrokontrolera lub generatora sygnału PWM, w stanie wysokim przyjmuje wartość 5V natomiast tego typu MOSFET'y wymagają napięcia dostarczanego na bramkę rzędu 9-12V. Przedstawione rozwiązanie układu mostka H staje się dużo bardziej skomplikowane od podstawowego układu mostka H. Większa komplikacja układu polega przede wszystkim na problemie zaprojektowania schematu elektrycznego oraz zbudowania działającego układu. Próby praktyczne wykazały, iż czasochłonne obliczenia, które miały na celu wyliczenie rezystancji oporników ustalających punkty pracy tranzystorów, nie przyniosły oczekiwanego rezultatu. Zbudowany układ podczas pracy okazał się pracować w odmienny sposób niż wynikało to z wyliczeń. Efekt odmiennej pracy układu był widoczny w momencie próby uruchomienia układu, gdy moc dostarczana na zaciski silnika stanowiła ok. 20% wyliczonej wartości. Problem ten pojawiał się przy pracy silnika w obu kierunkach. Ponowne obliczenia ujawniły błąd, popełniony

podczas wyliczania punktu pracy tranzystorów sterujących. Błąd został naprawiony, jednak układ w dalszym ciągu nie pracował w należyty sposób, mimo iż napięcie dostarczane na silnik było na odpowiednim poziomie, silnik podczas pracy w jedną stronę bardzo powoli nabierał prędkości obrotowej, natomiast przy pracy w stronę przeciwną pracował w sposób zakładany. Kolejne próby przeliczania wartości wszystkich rezystancji, jak i próby ponownej budowy układu, nie zaowocowały pracą układu, która była by zgodna z teoretycznymi wyznacznikami.

Po serii nieudanych prób uruchomienia układu sterującego pracą silnika postanowiono użyć gotowego układu mostka H. Gotowe do pracy układy dostępne są na rynku akcesorii dla modeli zdalnie sterowanych (modele RC) pod nazwą sterowniki silników bądź kontrolery prędkości. Dostępne są dwie wersje sterowników, jednokierunkowe, które pozwalają na obrót wału silnika tylko w jedna stronę, wykorzystywane głównie w modelach latających oraz pływających oraz sterowniki dwukierunkowe, pozwalające na obrót wału silnika w obie strony. Typowy kontroler prędkości silnika modelarskiego stanowi kompleksowy system sterowania silnikiem prądu stałego, zazwyczaj składa się z układu mostka H, układu mikroprocesorowego oraz przetwornicy napięcia. W tym przypadku mostek H jest wysterowany poprzez układ wzmacniaczy. Układ mikroprocesorowy ma na zadanie tłumaczenie pojedynczego sygnału PWM na szereg sygnałów ustalających prace silnika. Silnik jest zasilany napięciem pobieranym bezpośrednio ze źródła napięcia, natomiast układ mikroprocesorowy zasilany jest z wewnętrznej przetwornicy napięcia, która także może stanowić źródło zasilania dla urządzeń zewnętrznych, pracujących przy

Tabela 3.3-2. Charakterystyka kontrolera prędkości Hitec EZX-R.

Wymiary	45 x 33 x 15 mm
Waga	56 g
Napięcie operacyjne	7,2 - 8,4 V
Natężenie szczytowe (tryb jazdy "do przodu")	65 A
Natężenie maks. (tryb jazdy "do przodu")	46 A
Natężenie szczytowe (tryb jazdy "do tyłu")	32 A
Natężenie maks. (tryb jazdy "do tyłu")	23 A
Napięcie pracy (BEC)	5V +/- 0,2V 800mA
Zabezpieczenie termiczne	105C +/-10 ˚C
Oporność wewnętrzna	0,032 Ohm
Częstotliwość pracy	1,9 KHz
Ograniczenie silnika	23 zwoje

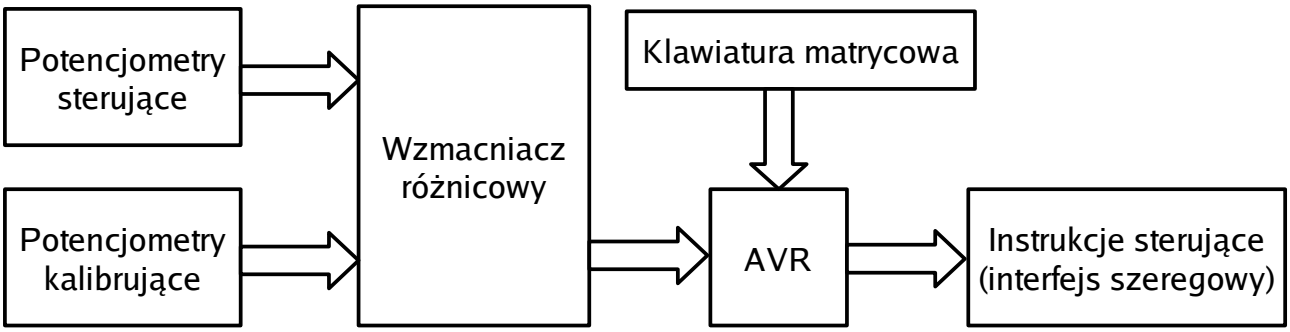
(http://www.hitecrcd.com/accessories/show?name=EZX-R)

napięciu 5V. Kontrolery prędkości mogą pracować pod różnym obciążeniem prądowym, w niektórych modelach dochodzącym nawet do 100A, jednak praca przy takim obciążeniu wymaga wydajnego źródła prądowego. Po szerszym zapoznaniu się z ofertą rynkową, zdecydowano się wybrać kontroler prędkości firmy Hitec model EZX-R. Charakteryzował się on, dwukierunkową pracą, obciążeniem prądowym wystarczającym do zasilenia silnika oraz niedużymi wymiarami. Dokładną charakterystykę użytego układu kontrolera przedstawia tabela 3.3-2. Wszystkie urządzenia firmy Hitec mają zestandaryzowane interfejsy wejściowe, dzięki temu stosując serwomechanizm oraz sterownik silnika tej firmy, upraszcza się cały system sterowania, ponieważ wystarczy zbudować jeden generator sygnału PWM do obsługi obydwu urządzeń.

Wiedząc już prawie wszystko o metodach sterowania układem napędowym oraz układem kierowniczym, przyszła kolej na budowę układu, który wyśle odpowiednie sygnały sterujące do układów jezdnych. Należało zatem zastanowić się nad najefektywniejszym sposobem rozwiązania tego problemu. Rozwiązaniem najbardziej optymalnym, a zarazem pierwszym, które zaczęto rozpatrywać stał się podstawowy układ mikrokontrolera AVR. Za rozwiązaniem tym przemawiały takie aspekty jak między innymi, zintegrowany licznik i układ czasowy, które można zaprogramować tak by generowały odpowiedni sygnał PWM oraz zintegrowany interfejs komunikacji szeregowej [11]. Budowany układ w praktyce będzie działał jako swego rodzaju tłumacz. Układ poprzez interfejs komunikacji szeregowej będzie otrzymywał dane, na podstawie których będzie generował odpowiednie sygnały sterujące. Jak się okaże w kolejnym rozdziale zastosowanie mikrokontrolera znacznie uprościło budowę układu tłumaczącego, w którym poza mikrokontrolerem znajduje się tylko kilka elementów pasywnych.

Zważywszy na wcześniej wymienione zalety mikrokontrolerów AVR, budowę manipulatora także postanowiono, w głównej mierze oprzeć na mikroukładzie AVR, w tym przypadku planowane jest wykorzystanie przetwornika analogowo-cyfrowego zintegrowanego w układzie AVR. Analogowy sterownik postanowiono zbudować w następujący sposób. W pierwszej kolejności należy zaadaptować analogowy joystick komputerowy, zadanie to wiąże się z koniecznością demontażu wszystkich zbędnych elementów elektronicznych istniejącego joysticka, powinny pozostać jedynie elementy mechaniczne. Następnym zadaniem jest zamocowanie nowych potencjometrów sterujących, w miejscu osi obrotu drążka oraz przepustnicy. Etapem końcowym budowy analogowego sterownika jest zbudowanie systemu elektronicznego, na który składać się będą następujące elementy: potencjometry sterujące, potencjometry kalibrujące, różnicowe wzmacniacze operacyjne, klawiatura matrycowa oraz mikrokontroler AVR. Jako układ wzmacniacza sygnału dostarczanego z potencjometrów sterujących, postanowiono użyć wzmacniacza operacyjnego pracującego w układzie różnicowym. Układ taki pozwolił na użycie

dodatkowych potencjometrów ustawiających współczynnik wzmocnienia sygnału wzmacniacza oraz zmieniających wielkość różnicy napięć pomiędzy wejściami wzmacniacza. Sygnał ze wzmacniaczy następnie jest podawany bezpośrednio na wyprowadzenia mikrokontrolera AVR, które kierują sygnał do wewnętrznego 10-bitowego przetwornika analogowo-cyfrowego. Tryb pracy przetwornika ADC jest ustawiany programowo, czyniąc go jednocześnie bardzo prostym i wygodnym w użyciu narzędziem pomiarowym. Kolejnym elementem układu jest klawiatura matrycowa w postaci przycisków umieszczonych w obudowie joysticka została ona włączona do projektu układu w momencie dostosowywania joysticka do potrzeb projektu. Klawiaturę postanowiono zachować ze względu na zwiększoną funkcjonalność analogowego sterownika.



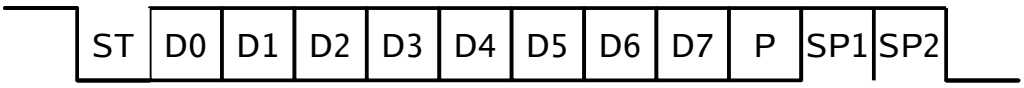
Rys. 3.3-4. Ogólny schemat budowy sterownika analogowego.

Możliwość dokładnej kalibracji joysticka, polegająca na takim ustawieniu parametrów pracy wzmacniacza tak aby skrajne położenia drążka generowały odpowiednio sygnał analogowy o maksymalnej i minimalnej wartości, wymusiła konieczność budowy układu, który będzie w sposób zrozumiały dla użytkownika wyświetlał wartość odczytanego sygnału. Układ postanowiono zbudować jako oddzielny moduł, który będzie opcjonalnie podłączany do systemu sterowania. Założono także, że układ będzie przechwytywał wszystkie dane transmitowane pomiędzy joystickiem a pojazdem, wyświetlając je na wyświetlaczu diodowym.

Ostatnim elementem jaki należało opracować aby system działał zgodnie z założeniami, jest zapewnienie komunikacji pomiędzy wszystkimi modułami systemu. Ze względu na chęć zapewnienia możliwości dowolnego łączenia modułów systemu postanowiono, że komunikacja będzie odbywać się w trybie szeregowym. Wybór takiego rodzaju transmisji był oczywisty, ponieważ mikrokontrolery stanowiące podstawę układów elektronicznych joysticka, sterownika układów jezdnych oraz moduł wyświetlacza, zostały wyposażone w układ transmisji szeregowej

USART. Natomiast komputer PC oraz minikomputer ARM są wyposażone w porty szeregowe RS232. Układ USART czyli uniwersalny synchroniczny i asynchroniczny odbiornik i nadajnik, jest to specjalny układ który tłumaczy bajty danych (znaki) na ciągi bitów i na odwrót. Układ stanowiący port RS232 zezwala tylko na prace w trybie asynchronicznym, dlatego transmisja będzie odbywać się w tym właśnie trybie.

Transmisja asynchroniczna jest to metoda szeregowego przekazywania danych znak po znaku, które są kodowane jako ciągi bitów oddzielanych specjalnymi znacznikami rozdzielającymi, początku i końca znaku. Transmisja nie jest synchronizowana żadnymi znakami specjalnymi czy ciągami synchronizującymi. Początek transmisji jest określany jako zmiana wartości początkowej z poziomu wysokiego na niski. Początek transmisji każdego ze znaków jest określany z chwilą rozpoczęcia jego nadawania. Interpretacja znaku w odbiorniku następuje po nadaniu kilku bitów danych, dlatego niewielkie różnice częstotliwości taktowania, po obu stronach linii transmisyjnej, nie maja istotnego znaczenia. Dla detekcji błędów transmisji używa się jednobitowego znacznika parzystej liczby takich samych wartości bitów jednego znaku. Dane są transmitowane na jednym przewodzie transmisyjnym dlatego w układach transmisji danych każdy bit musi być rozpoznany w czasie określonym przez takt zegara synchronizującego. Ciąg bitów określających znak jest transmitowany w znormalizowanej formie zwanej ramką (Rys. 3.3-5.) [5].



Rys. 3.3-5. Format danych dla transmisji asynchronicznej [5].

Pomimo iż interfejsy USART oraz RS232 można zaprogramować tak by używały tego samego trybu transmisji, niemożliwe jest bezpośrednie połączenie wyjść układu USART i portu RS232. Spowodowane jest to różnym poziomem napięć na wyjściach układów, w RS232 napięcia wynoszą odpowiednio, stan wysoki -12V, stan niski +12V, natomiast w USART odpowiednio, stan wysoki 5V, stan niski 0V. Konieczne jest więc wykorzystanie układu dopasowującego poziomy napięć.

4. Elektronika pojazdu

4.1. Zasilanie

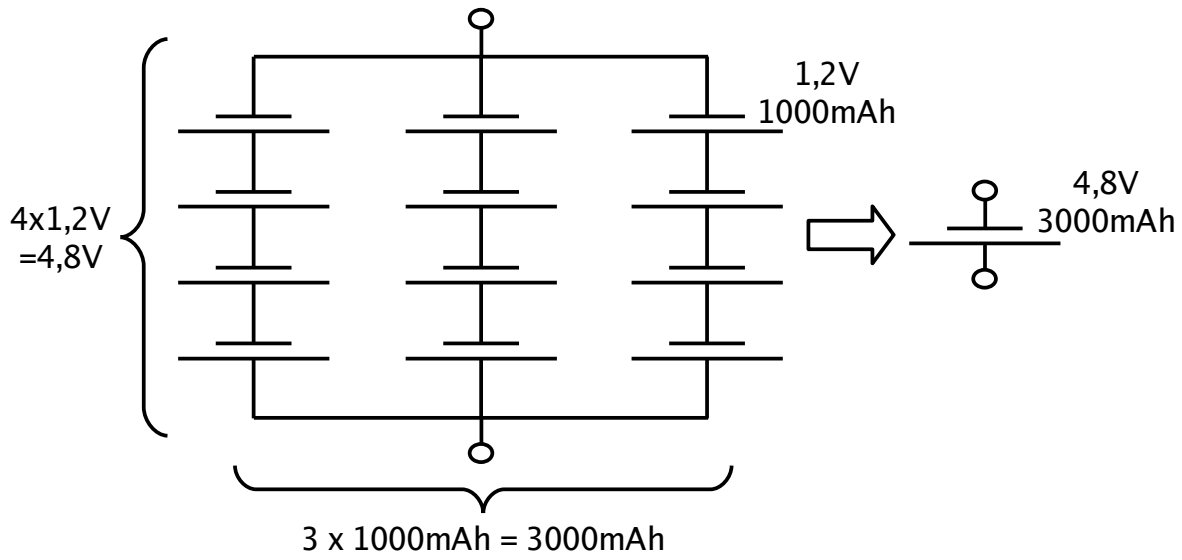
System sterowania pojazdem docelowo ma być systemem bezprzewodowym, oznacza to że układ zasilania musi zostać podzielony na dwie niezależne części. Jedna część układu zasilającego będzie miała za zadanie zasilenie wszystkich układów elektronicznych zainstalowanych na podwoziu, druga zaś będzie zasilać układy podłączone do komputera. Układy mikroprocesorowe oraz serwomechanizm wymagają zasilania napięciem stałym o wartości około 5V, natomiast silnik wymaga większego napięcia, kształtującego się na poziomie ok. 7-14V. Część układu zasilania, zaopatrującego w prąd układy podłączone do komputera, jest najprostsza do zrealizowania ponieważ do komputera będzie podłączony jedynie mikroprocesorowy joystick, który potrzebuje pojedynczego napięcia. Zakładając, że joystick będzie pobierał nieduże iloci prądu, postanowiono wykorzystać jako źródło zasilania dwie linie portu USB komputera. Napięcie na porcie jest napięciem stabilizowanym oraz stałym, w zależności od komputera napięcie kształtuje się w zakresie od 4,8 do 5,2V. Obciążalność prądowa portu mieści się w zakresie 100-500mA.

Tabela 4.1-1. Zasilanie poszczególnych układów pojazdu.

Rodzaj układu/urządzenia	Napięcie	Natężenie
Silnik	7-14V	maks. 20A
Serwomechanizm	4,5-6V	ok. 100mA
Sterownik silnika	7,2V lub 8,4V	w zależności od silnika, maks. 65A
mikrokontroler	4,5-6V	maks. 300mA
mikrokomputer ARM	5V	maks. 1A
karta sieciowa Wi-Fi	5V	maks. 300mA

W przypadku układów elektronicznych pojazdu kwestia zasilania jest bardziej skomplikowana, ponieważ na elektronikę podwozia składa się kilka układów i urządzeń, z których każde wymaga nieco odmiennych parametrów prądowo-napięciowych. W tabeli 4.1-1. przestawiono zestawienie obrazujące wymagania stawiane układowi zasilania przez poszczególne urządzenia. Ze względu na mobilność pojazdu źródłem napięcia będzie akumulator, który zapewnia stałe napięcie. Akumulator powinien posiadać odpowiednią wydajność prądową, aby przy pracy silnika nie było spadków napięcia lub innych zakłóceń na linii zasilania. Jest to szczególnie ważne gdyż wszelakie zakłócenia na linii zasilania mogłyby skutkować różnymi nieprzewidywalnymi błędami w pracy układów cyfrowych. W celu jak najlepszego odseparowania zakłóceń pojawiających się na linii zasilania każdy układ elektroniczny będzie wyposażony w filtr RC. Użycie filtrów jest szczególnie ważne w przypadku gdy układy są zasilane z tego samego źródła co urządzenia dużej mocy (np. silnik), które w momencie włączania/wyłączania generują zakłócenia.

Dobór akumulatora był jednym z poważniejszych zadań, od jego wyboru zależała sprawność całego projektu. Pierwszym pomysłem było zastosowanie baterii zbudowanej ze standardowych akumulatorków typu AA (R6). Rozwiązanie to niesie ze sobą korzyści takie jak, niska cena baterii, niewielka masa źródła zasilania oraz długa żywotność całego kompletu. Dodatkowym atutem tego rozwiązania jest możliwość wymiany pojedynczego akumulatorka, zamiast wymiany całej baterii, w razie awarii, co zmniejsza koszty ewentualnych napraw. Jednak ograniczeniem które wyeliminowało to rozwiązanie była bardzo mała wydajność prądowa zestawu zbudowanego z połączonych szeregowo i równolegle akumulatorów AA.



Rys. 4.1-1. Schemat budowy baterii na przykładzie połączenia 12 akumulatorów.

Następnie uwagę postanowiono skupić na akumulatorach żelowych, z pominięciem akumulatorów ołowiowych ponieważ są bardziej niebezpieczne w użytkowaniu od akumulatorów żelowych. Największa zaletą akumulatorów żelowych jest ich pojemność oraz brak tak zwanego efektu pamięci, drastycznie zmniejszającego żywotność akumulatora w przypadku ładowania nie w pełni rozładowanego akumulatora. Dodatkową zaletą tego typu źródła zasilania jest wysoka wydajność prądowa, która umożliwiwiłaby poprawną prace silnika oraz pozostałych układów pojazdu. Jednak mimo dobrych parametrów prądowych oraz niedużej masy, rozwiązanie to postanowiono odrzucić ze względu na duże rozmiary akumulatora, które znacznie utrudniałoby zwarty montaż wszystkich urządzeń na podwoziu. Rozwiązaniem które okazało się optymalne dla opracowywanego projektu było użycie wysokonapięciowego pakietu RC-pak, stosowanego do zasilania modeli zdalnie sterowanych. Pakiet taki składa się z akumulatorów wykonanych w technologii niklowo-wodorkowej (Ni-Mh). Pakiet taki łączy zalety baterii akumulatorków AA oraz akumulatora żelowego, takie jak mała masa, wysoka obciążalność prądowa oraz brak efektu pamięci.

Rys. 4.1-2. Typy akumulatorów.
a) Bateria akumulatorów typu AA, połączenie szeregowe.
b) Przykład akumulatora żelowego.
c) Pakiet RC-pak

Z pośród wszystkich dostępnych na rynku modeli akumulatorów wybrano model: RC-Pak Profitexx 4000mAh. Jest to jeden z najwydajniejszych pakietów dostępnych na rynku w momencie prowadzenia prac projektowych. Dokładną charakterystykę użytego pakietu przedstawia tabela 4.1-2 (Dane pobrane z http://www.profitexx.com.pl/). Dodatkowo akumulator posiada możliwość ładowania go dużym prądem co sprawia, że kompletne naładowanie wyładowanych akumulatorów zajmuje około dwie-trzy godziny. Pomimo iż akumulator charakteryzuje się dużą wydajnością postanowiono pojazd zaopatrzyć w dwa równolegle połączone pakiety, ze względu na zwiększenie długości czasu pracy oraz większą stabilność napięciowo-prądową na linii zasilania.

Tabela 4.1-2. Charakterystyka użytego akumulatora.

Model:	RC Pak Profitexx
Pojemność:	4000 mAh
Napięcie znamionowe:	7,2 V
Maks. prąd obciążenia	48A
Waga:	380 gram
Wymiary:	13,5cm x 2,3cm x 4,5cm
Złącze:	standard Tamiya
Temperatura pracy	do 105°C

Zastosowane akumulatory są zasadniczo bez obsługowe jednak przy ich użytkowaniu należy przestrzegać następujących zasad:

- Akumulator można doładowywać w dowolnym momencie, jednak radzi się ładować pakiety dopiero w momencie całkowitego ich rozładowania.
- Do ładowania akumulatorów należy używać wyłącznie załączonej automatycznej ładowarki
- Zaleca się po zakończeniu ładowania (zapalona zielona dioda na ładowarce) odłączyć przewód ładowarki oraz odłączyć ładowarkę od sieci elektrycznej
- NIE WOLNO dopuścić do zwarcia styków akumulatora
- Nie należy podłączać akumulatorów w odwrotnej polaryzacji

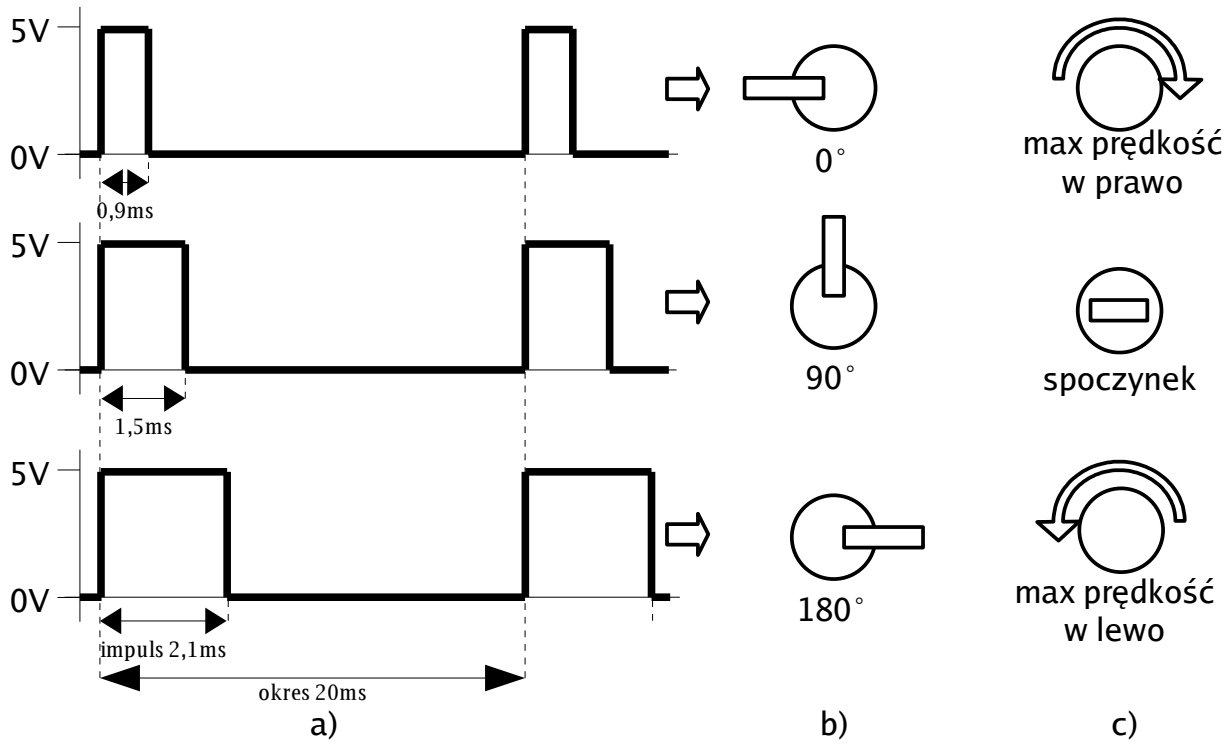
Podczas poszukiwań sterownika silnika okazało się, że wybrany model może stanowić źródło zasilania dla pozostałych zewnętrznych układów. Napięcie udostępniane przez sterownik jest stabilizowane oraz kształtuje się na poziomie 5V. Rozwiązanie to znacznie upraszcza strukturę układu zasilania, gdyż zbędne okazuje się korzystanie z dodatkowych stabilizatorów napięcia 5V dla układów cyfrowych. W efekcie takiego działania można wyprowadzić linię zasilania bezpośrednio ze sterownika silnika.

4.2. Sterowanie napędem

Podstawowymi elementami wykonawczymi sterowania podwozia są:

- serwomechanizm odpowiedzialny za ustawianie kąta wychylenia kół
- silnik dużej mocy wraz z regulatorem napięcia

Elementy te wymagają sterowania ściśle określonym sygnałem PWM. Ponieważ oba urządzenia wykonane są przez tego samego producenta, do poprawnej pracy wymagają sygnału PWM o takich samych parametrach. Czyniąc elektroniczny układ sterowania znacznie prostszym w projektowaniu oraz budowie. Te same sygnały sterujące mają jednak różną interpretacje ze względu na różne funkcje mechaniczne urządzeń wykonawczych. Serwomechanizm interpretuje odebrany sygnał jako zadany bezwzględny kąt wychylenia, natomiast sterownik silnika interpretuje sygnał jako kierunek i prędkość obrotową wału silnika. Charakterystyka sygnałów sterujących wraz z interpretacjami dla poszczególnych urządzeń wykonawczych przedstawia Rys. 4.2-1. [3].

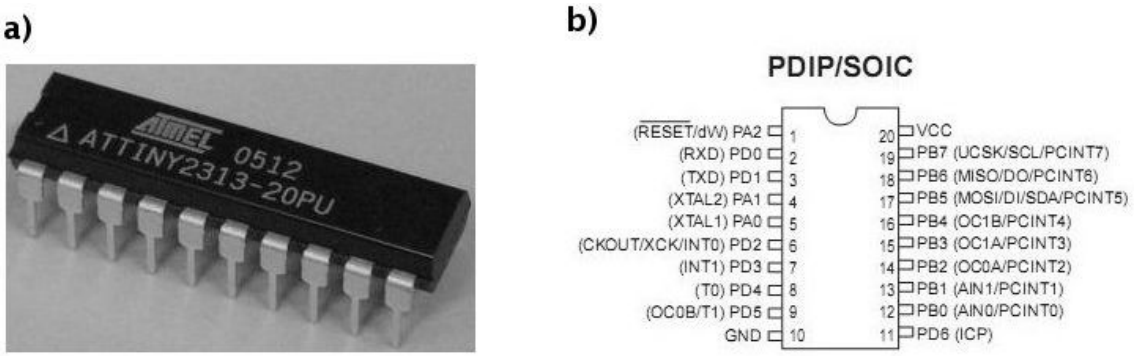


Rys. 4.2-1. Charakterystyka sygnałów sterujących.
a) Sygnał sterujący.
b) Interpretacja dla serwomechanizmu.
c) Interpretacja dla sterownika silnika.

4.2.1. Charakterystyka modułu

Generator sygnałów PWM stanowi moduł, którego budowa została oparta na mikrokontrolerze Atmel ATtiny2313. Moduł pobiera informacje sterujące z portu szeregowego a następnie po odpowiednim ich przetworzeniu, wysyła sygnały PWM na odpowiednie linie sterujące urządzeniami wykonawczymi. Podczas budowy układu przewidziane zostały możliwości rozbudowy pojazdu. Pomimo faktycznego zapotrzebowania na dwa generatory sygnału PWM układ jest w stanie sterować czterema urządzeniami. W chwili obecnej dwie dodatkowe linie sygnałowe zostały wykorzystane do sterowania oświetleniem diodowym pojazdu, które stanowią przednie i tylne światła. Światła nie stanowią żadnego funkcjonalnie ważnego elementy projektu, zatem mogą być zastąpione dowolnym urządzeniem sterowanym poprzez sygnał PWM. Tym samym podłączenie dodatkowych urządzeń wykonawczych nie będzie stwarzać dodatkowych problemów natury technicznej, ponieważ w pełni funkcjonalne wyjścia zostały już fizycznie i programowo zaimplementowane w układzie.

Moduł sterujący został stworzony w celu zapewnienia interfejsu komunikacji zgodnego z RS232 na poziomie programowym, ze względu na wcześniejsze założenia projektowe dotyczące modularności całego systemu zapewniającej możliwość dowolnej konfiguracji systemu sterowania. Zastosowane rozwiązanie jednocześnie zabezpiecza przed problemami transmisji, które mogłyby wystąpić podczas przesyłania wielu sygnałów PWM po magistrali równoległej. Sygnały te podczas transmisji mogłyby się wzajemnie zakłócać, natomiast podczas transmisji szeregowej i późniejszej konwersji na osobne sygnały problem ten nie występuje.



Rys. 4.2.1-1. Mikrokontroler ATtiny2313 [12].
a) Zdjęcie.
b) Rozkład wyprowadzeń.

4.2.2. Charakterystyka zastosowanego układu

Układ dzięki mikroprocesorowi ATtiny2313 charakteryzuje się dużą wydajnością obliczeniową, małym poborem prądu oraz odpowiednią ilością zintegrowanych w układzie interfejsów peryferyjnych. Parametry te doskonale spełniają wymagania stawiane przed układem.

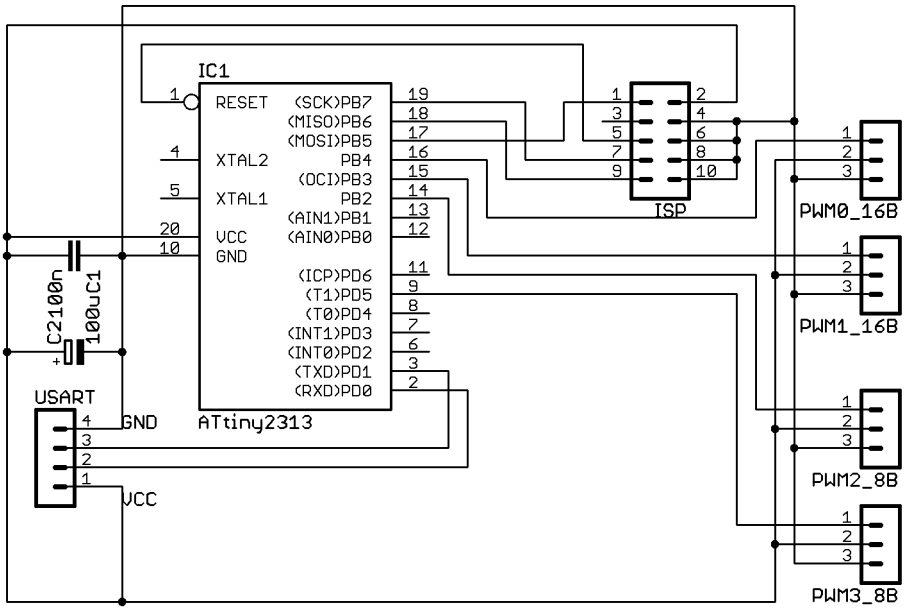
Lista najważniejszych parametrów mikrokontrolera przedstawia się w następujący sposób:

- Wykonanie w technologii CMOS
- Architektura 8-bitowa
- 120 instrukcji procesora – większość zajmuje jeden cykl zegarowy
- 32 jedno-bajtowe rejestry robocze
- Moc obliczeniowa do 1 MIPS na każdy 1 MHz
- 2KB dostępnej pamięci programu typu FLASH
- 128B dostępnej pamięci danych typu EEPROM
- 128B dostępnej pamięci SRAM
- Peryferia
  - pojedynczy 8-bitowy licznik/układ czasowy
  - pojedynczy 16-bitowy licznik/układ czasowy
  - 4 kanały PWM
  - USART z możliwością transmisji Full Duplex
  - SPI zapewniający możliwość programowania ISP
- Obsługa wewnętrznych i zewnętrznych źródeł przerwań
- Wewnętrzny generator sygnału zegarowego
- Napięcie zasilania od 2,7V do 5,5V
- Taktowanie zegarem do 20MHz przy zasilaniu min. 4,5V

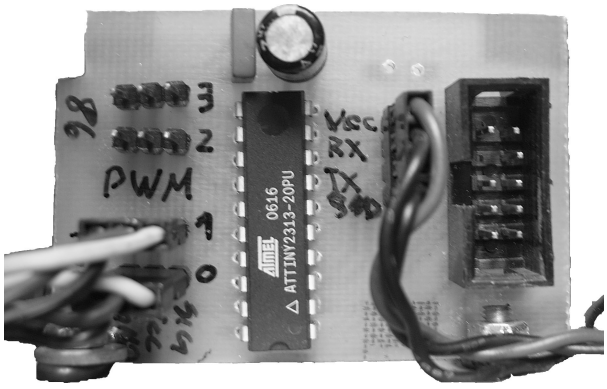
Najbardziej znaczącymi cechami tego mikrokontrolera, ze względu na zastosowanie są, cztery kanały PWM wraz z układami czasowymi oraz interfejs USART. Duża możliwa moc obliczeniowa zapewnia przetworzenie nowo odebranych informacji przed upływem pełnego cyklu sygnału PWM. Zegar systemowy ustawiony na 8MHz pozwala na wykonanie ok. 160 tysięcy operacji w trakcie trwania pojedynczego cyklu sygnału PWM, co zapewnia wystarczającą ilość czasu potrzebnego na odebranie, przetworzenie danych oraz zmianę parametrów sygnału PWM bez jego zniekształceń.

4.2.3. Opis budowy modułu

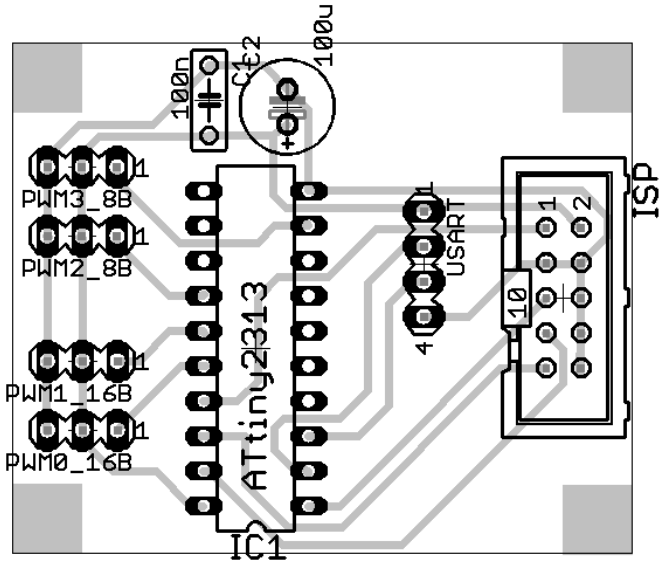
Szczegóły budowy układu sterującego przedstawiają poniższe schematy.



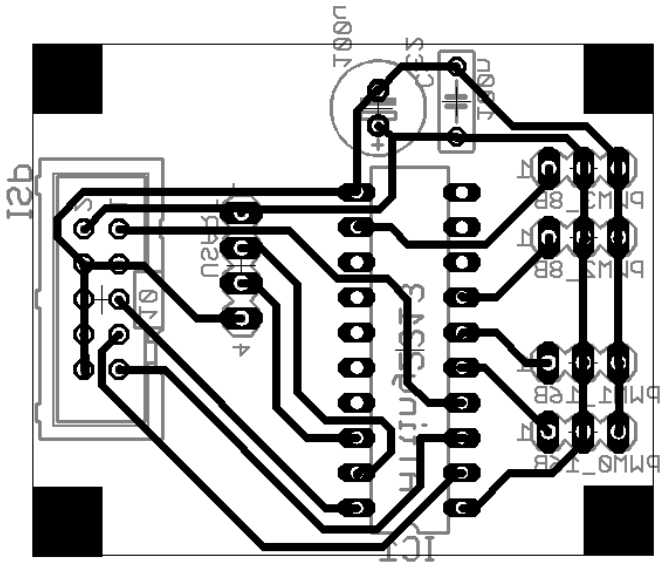
Rys. 4.2.3-1. Schemat ideowy układu



Rys. 4.2.3-2. Zdjęcie zmontowanego modułu.



Rys. 4.2.3-3. Płytka drukowana – strona układów.



Rys. 4.2.3-4. Płytka drukowana – strona ścieżek.

4.2.4. Opis działania modułu

Pierwszą czynnością jaką wykonuje układ zaraz po włączeniu i inicjalizacji zmiennych odpowiadających za poprawną prace układu jest zsynchronizowanie się z układem nadającym dane. Czynność ta jest niezaprzeczalnie ważna ze względu na fakt iż przesyłane są wielobajtowe pakiety danych, które składają się z szeregu ramek transmisyjnych, z których każda zawiera jednobajtową informacje. Brak synchronizacji może doprowadzić do odebrania poszczególnych ramek z niewłaściwej kolejności, czego skutkiem byłaby błędna praca układu. Dzięki oczekiwaniu na synchronizację dodatkowo zapewnione jest zabezpieczenie przed uruchomieniem napędu zanim zostanie wysłana poprawna paczka danych zawierająca informacje sterujące. Zabezpieczenie to jednocześnie chroni elementy mechaniczne podwozia przed fizycznymi uszkodzeniami wynikającymi z nieprzewidywalności zachowania się pojazdu, w przypadku pojawienia się tego typu błędu. Po inicjalizacji układu stan wyjść PWM odpowiednio dla silnika i serwomechanizmu jest ustawiany na postój i wycentrowanie kół. Następnie układ przechodzi w stan oczekiwania dopóki nie odnotuje poprawnej sekwencji synchronizującej. Po poprawnym odebraniu i zinterpretowaniu paczki synchronizującej układ przechodzi do kolejnego, docelowego trybu pracy, podczas którego odbiera paczkę danych i przepisuje ją do tablicy. W efekcie tej operacji w każdej jednobajtowej komórce tablicy mamy zapisane informacje dotyczące poszczególnych wartości wysyłanych przez układ sterujący. W końcowej fazie pętli głównej programu aktualizowane są wartości rejestrów odpowiedzialnych za charakterystykę przebiegów na wyjściach PWM.

Pierwszy bajt	Drugi bajt	Trzeci bajt
0xFA	0xFE	0xFB

Rys. 4.2.4-1. Paczka synchronizująca.

Paczka synchronizująca składa z trzech kolejnych ramek transmisyjnych (rys 4.2.4-1). Wartości kolejnych bajtów zostały dobrane w taki sposób aby były różne od dowolnej informacji wysyłanej w pojedynczej ramce danych. Co oznacza, że dowolna poprawnie odebrana, bez wcześniejszej synchronizacji, paczka danych nie jest w stanie wyprowadzić modułu ze stanu

oczekiwania. Algorytm synchronizujący układ (Rys 4.2.4-4.) został opracowany w taki sposób, że w przypadku odebrania innej wartości niż przewidziane podczas procesu synchronizacji, układ powraca na początek tego procesu. Pomimo zalet przedstawionego algorytmu posiada on również wady. W przypadku błędu i utraty synchronizacji program nie pozwala po raz kolejny zsynchronizować nadajnika i odbiornika. Konieczne w tym przypadku jest zresetowanie modułu poprzez ponowne uruchomienie układu.

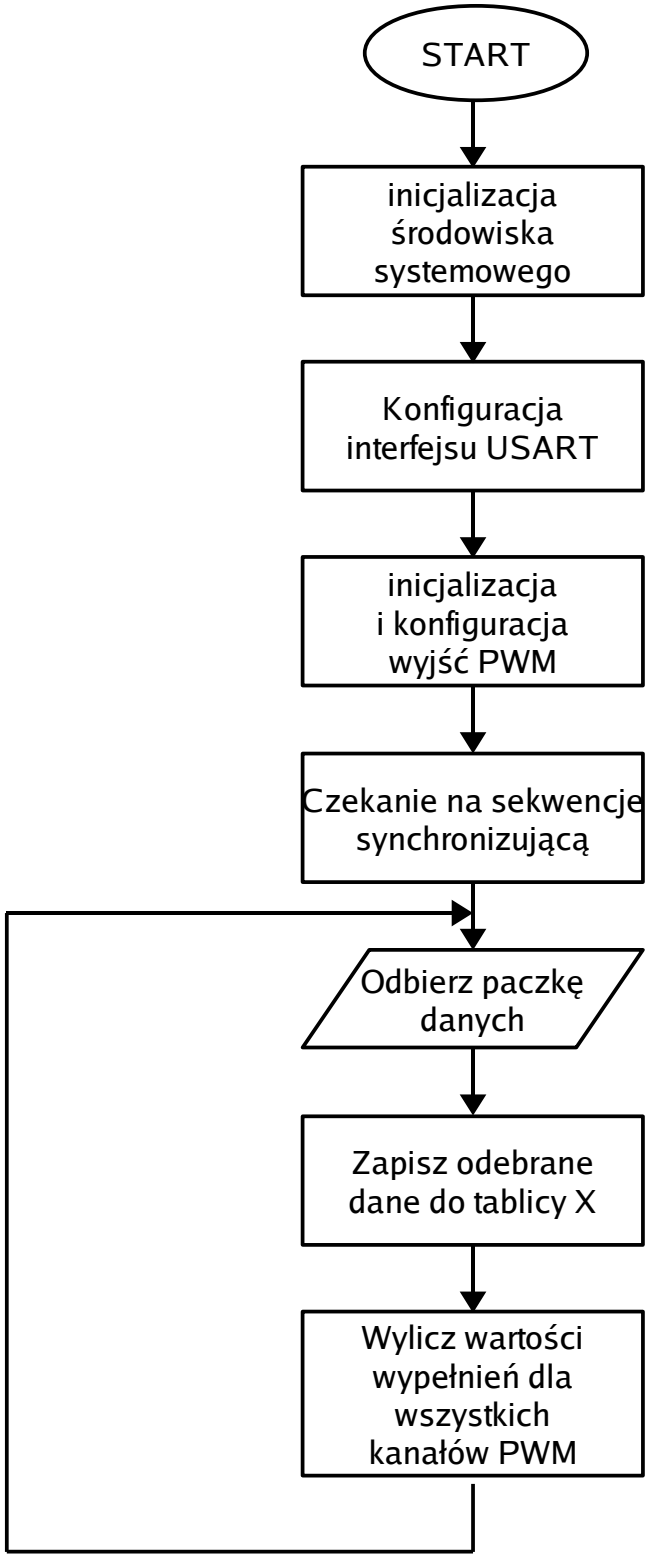
Pierwszy bajt	Drugi bajt	Trzeci bajt	Czwarty bajt
prędkość pojazdu w przód/w tył	wychylenie kół w lewo/w prawo	prędkość maksymalna	funkcje dodatkowe

Rys. 4.2.4-2. Paczka danych.

Na rysunku 4.2.4-2 przedstawiony został format wykorzystywanego pakietu danych. Na pakiet składają się cztery ramki transmisyjne, które przenosza łącznie cztery bajty informacji. Każdy odebrany bajt danych przenosi informację odczytaną z poszczególnych czujników manipulatora. Wartości poszczególnych bajtów danych kolejno interpretowane są jako:

- kierunek jazdy oraz stosunek prędkości zadanej do maksymalnej
- kąt wychylenia kół
- dopuszczalna prędkość maksymalna pojazdu
- stan dodatkowych możliwych do zamontowania urządzeń

Wartości trzech pierwszych bajtów moją się jedynie zmieniać w zakresie od 0 do 200, co wynika z konieczności ich przeskalowania. Zastosowany okrojony zakres możliwych wartości znacznie upraszcza przeskalowywanie odebranych danych, na wartości konieczne do ustawienia odpowiedniej wartości wypełnienia sygnału PWM. Optymalizacja ta polega na wykorzystaniu dużo mniejszej ilości rozkazów procesora, tym samym sprawiając, że proces przeskalowywania trwa bardzo krótko. Bity ostatniego bajtu paczki, nie są przeskalowywane i każdy z nich służy bezpośrednio do sterowania przyszłymi funkcjami pojazdu. W chwili obecnej dwa bity z tej przestrzeni zostały wykorzystane do sterowania oświetleniem zamontowanym w pojeździe.



Rys. 4.2.4-3. Algorytm programu mikrokontrolera.

Przedefiniowanie nazw rejestrów w celu większej przejrzystości kodu.

```
#define SERVO OCR1A
#define MOTOR OCR1B
#define FLIGHT OCR0A
#define BLIGHT OCR0B
```

Włączenie oraz skonfigurowanie interfejsu komunikacji szeregowej.

```
void USART_init(void)
{
  UBRRH = 0x00;      // Starszy bajt ustawiający prędkość transmisji.
  UBRRL = 0x33;      // Młodszy bajt ustawiający prędkość transmisji.
  UCSRB = 0x18;      // Włączenie nadajnika oraz odbiornika.
  UCSRC = 0x8E;      //Ustawienie trybu pracy na asynchroniczny, ustawienie formatu
ramki                // na 8 bitów danych, wyłączenie parzystości oraz dwa bity stopu.
}
```

Odbiór z portu jednego bajtu danych.

```
uint8_t USART_receive(void)
{
  while( !(UCSRA & 0x80) );
  return UDR;
}
```

Skonfigurowanie wyjść układu do generowania sygnałów PWM o 16-bitowej rozdzielczości.

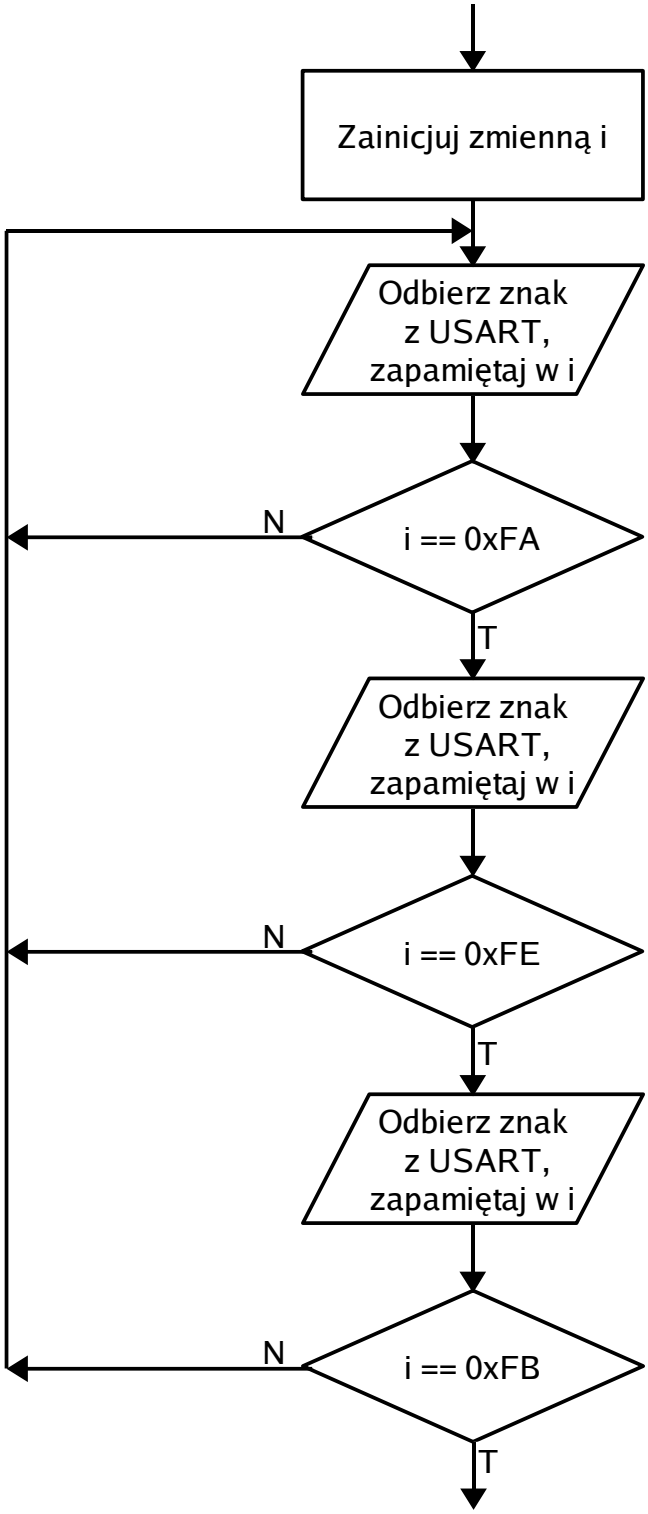
```
void PWM16_init(void)
{
  DDRB|=0x18;        // Ustawienie portów jako wyjście.
  PORTB|=0x18;       // Ustawienie stanu początkowego na stan wysoki.

  TCCR1A=0xA0;       // Podpięcie do portów wyjścia układu czasowego.
  TCCR1B=0x12;       // Włączenie trybu generacji fali typu PWM, ustawienie częstotliwości
                     // wyjściowego sygnału.
  ICR1=0x2710;       // Wartość maksymalna do jakiej zlicza licznik.
  SERVO=750;         // Ustawienie domyślnego wypełnienia dla pierwszego wyjścia.
  MOTOR=750;         // Ustawienie domyślnego wypełnienia dla drugiego wyjścia.
}
```

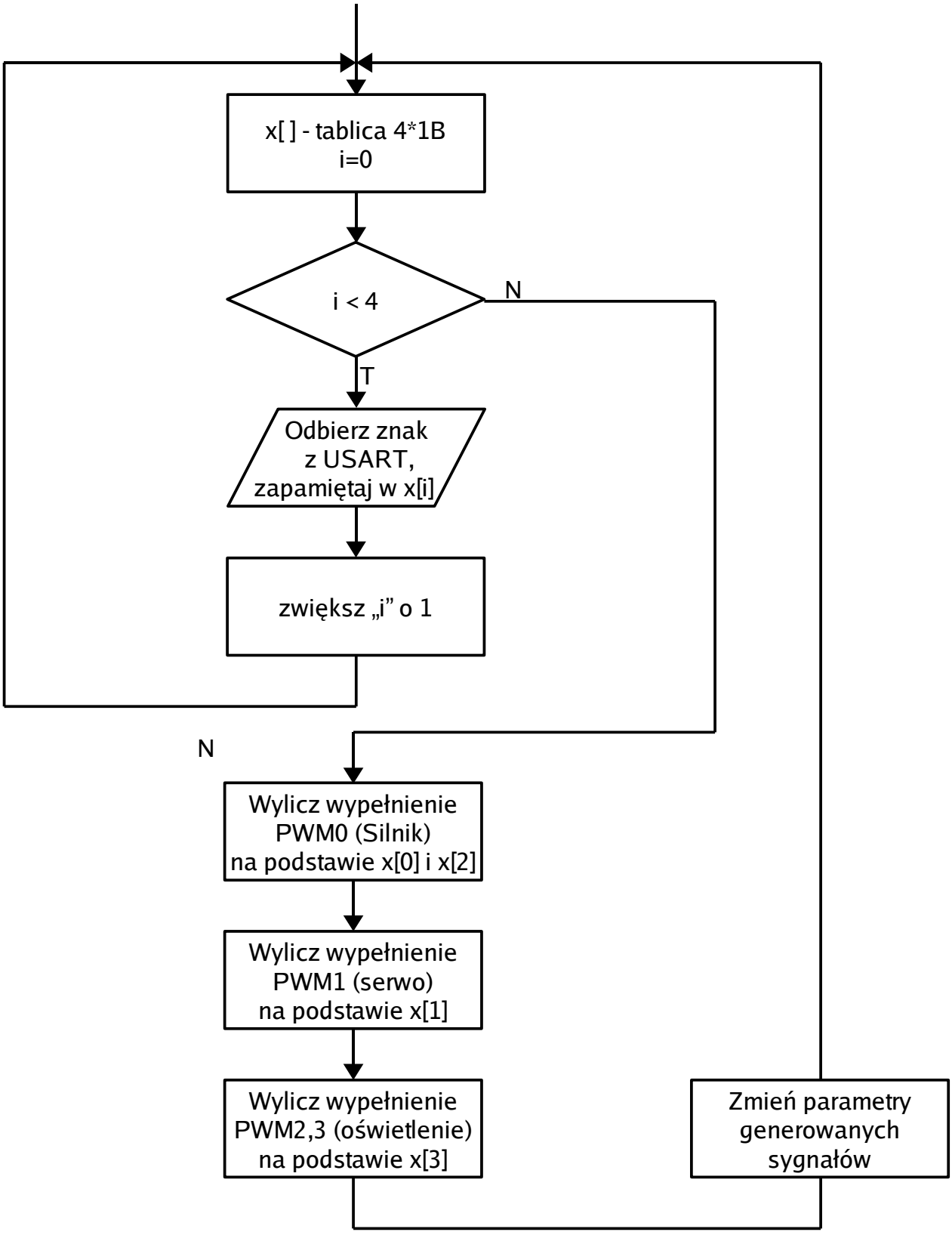
Skonfigurowanie wyjść układu do generowania sygnałów PWM o 8-bitowej rozdzielczości

```
void PWM8_init(void)
{
  DDRB|=0x04;
  PORTB|=0x04;
  DDRD|=0x20;
  PORTD|=0x20;

  TCCR0A=0xA1;
  TCCR0B=0x04;
  FLIGHT=0x00;
  BLIGHT=0x00;
}
```



Rys. 4.2.4-4. Algorytm procesu synchronizacji



Rys. 4.2.4-5. Algorytm odebrania i przetworzenia paczki

Część programu odpowiedzialna za synchronizacje układu.

```
do
{
  do
  {
    do
      i=USART_receive();
    while(i!=0xFA);
    i=USART_receive();
  }
  while(i!=0xFE);
  i=USART_receive();
}
while(i!=0xFB);
```

Główna pętla programu.

```
while(1)
{
```

Odebranie pakietu danych i zapisanie go do tablicy.

```
  for(i=0;i<4;i++)
    x[i]=USART_receive();
```

Przeskalowanie odebranej wartości oraz ustawienie poziomu wypełnienia sygnału sterującego silnikiem.

```
  if(x[0]>=100)
    x[0]=(((uint16_t)(x[0]-100)*x[2])/200)+100;
  else
    x[0]=100-(((uint16_t)(100-x[0])*x[2])/200);
  pwm16=((uint16_t)x[0]*3)+450;
  MOTOR=pwm16;
```

Przeskalowanie odebranej wartości oraz ustawienie poziomu wypełnia sygnału sterującego serwomechanizmem.

```
  pwm16=1050-((uint16_t)x[1]*3);
  SERVO=pwm16;
```

Ustawienie stanu oświetlenia na poziomie pojedynczych bitów, ostatniego bajtu paczki danych

```
  if(x[3]&0x08)
    FLIGHT=0xFF;
  else
    FLIGHT=0x00;
  if(x[3]&0x04)
    BLIGHT=0xFF;
  else
    BLIGHT=0x00;
}
```

4.3. Linijka diodowa

4.3.1. Charakterystyka modułu

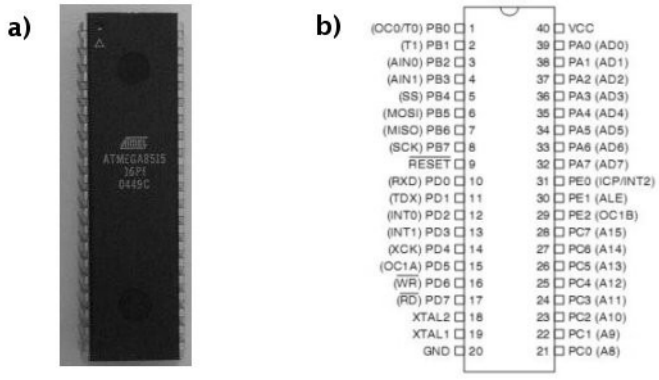
Moduł linijki diodowej został opracowany w celu wyświetlenia zawartości ramek transmisyjnych przesyłanych na linii łączącej moduł generatora sygnałów PWM z nadajnikiem informacji sterujących. Obecne oprogramowanie sprawia, że moduł jest wykorzystywany jako urządzenie umożliwiające debugging układów komunikujących się przez wykorzystany w projekcie systemu interfejs szeregowy, wyświetlając zawartość każdej lub tylko co niektórych z odebranych ramek transmisyjnych. Zawartość ramek jest wyświetlana w trybie binarnym na wyświetlaczu składającym się z szeregu ośmiu diod LED, a wybór danych do wyświetlenia następuje za pomocą mikro-przełączników. Budowa modułu została oparta na mikrokontrolerze Atmel Atmega8515. Zastosowanie takiego układu pozwoliło na zbudowanie uniwersalnego urządzenia kontrolującego stany na poszczególnych interfejsach. Zastosowanie tego mikrokontrolera umożliwia przedstawienie w czytelny dla użytkownika sposób informacji o stanach wielu linii przesyłowych, przede wszystkim stanu na linii wyjściowej, portu szeregowego, układu nadawczego. Układ scalony Atmega8515 jest wyposażony w cztery uniwersalne porty 8-bitowe oraz jeden port 3-bitowy, które można skonfigurować do pracy z różnymi typami sygnałów w tym analogowych przy zastosowaniu różnych konfiguracji układów peryferyjnych wewnętrznych oraz zewnętrznych. Podczas budowy modułu przestrzegano zasady „maksymalnej” prostoty późniejszej rozbudowy, dlatego wszystkie porty zostały wyprowadzone na uniwersalnych złączach typu GoldPin. Układ został tak zaprojektowany i zbudowany, aby nie zakłócać transmisji danych pomiędzy układami. Takie podejście pozwala na śledzenie przebiegów sygnałów bez ingerencji w układy generujące te sygnały. Dlatego układ nie będzie źródłem problemów podczas rozbudowy pojazdu o dodatkowe urządzenia. Dodatkowym atutem jest to, że układ można w prosty sposób wykorzystać w celu dalszej rozbudowy systemu sterowania, a zwłaszcza jako prototyp kolejnego modułu, przy odpowiednim jego zaprogramowaniu. Zastosowanie takich rozwiązań jest zgodne z przyjętą na etapie początkowych założeń, zasadą modularności.

4.3.2. Charakterystyka zastosowanego układu

Układ dzięki mikroprocesorowi ATmega8515 charakteryzuje się dużą ilością zintegrowanych w układzie interfejsów wejścia/wyjścia, do których można podłączyć źródła lub odbiorniki danych.

Lista najważniejszych parametrów mikrokontrolera przedstawia się w następujący sposób:

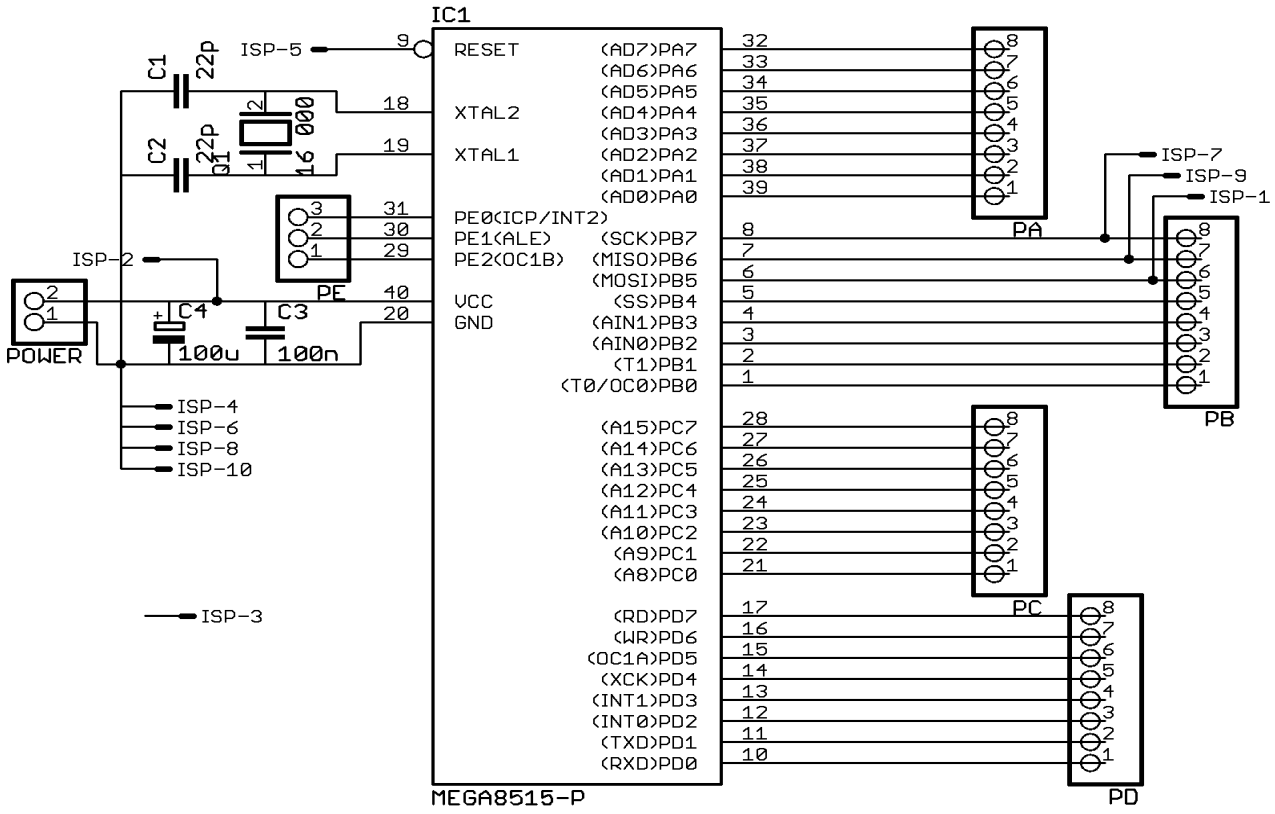
- Architektura 8-bitowa wykonana w technologii CMOS
- 130 instrukcji procesora – większość zajmuje jeden cykl zegarowy
- 32 jedno-bajtowe rejestry robocze
- 8KB dostępnej pamięci programu typu FLASH
- po 512B dostępnej pamięci danych typu EEPROM oraz SRAM
- cztery 8-bitowe oraz jeden 3-bitowy uniwersalny port wejścia/wyjścia
- Peryferia
  - pojedynczy 8-bitowy licznik/układ czasowy
  - pojedynczy 16-bitowy licznik/układ czasowy
  - trzy kanały PWM
  - USART
  - SPI zapewniający możliwość programowania ISP
  - komparator analogowy
- Obsługa wewnętrznych i zewnętrznych źródeł przerwań
- Wewnętrzny generator sygnału zegarowego
- Taktowanie zegarem do 16MHz przy napięciu zasilania od 4,5V do 5,5V



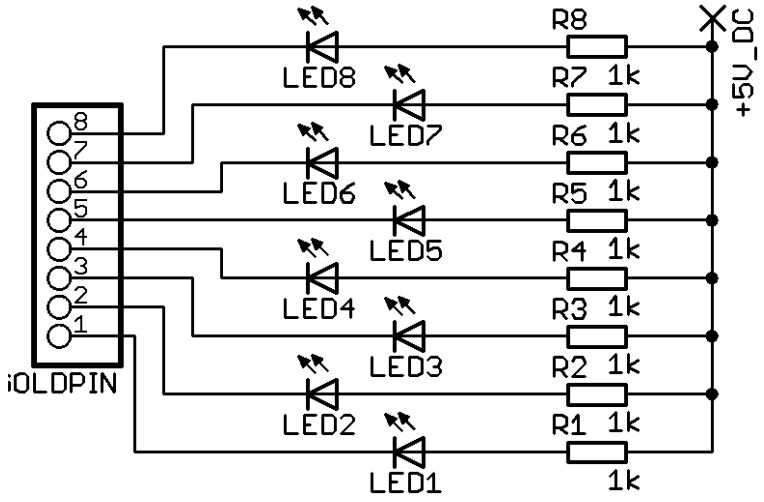
Rys. 4.3.2-1. Mikrokontroler Atmega8515 [12]
a) Zdjęcie. b) Rozkład wyprowadzeń.

4.3.3. Opis budowy modułu

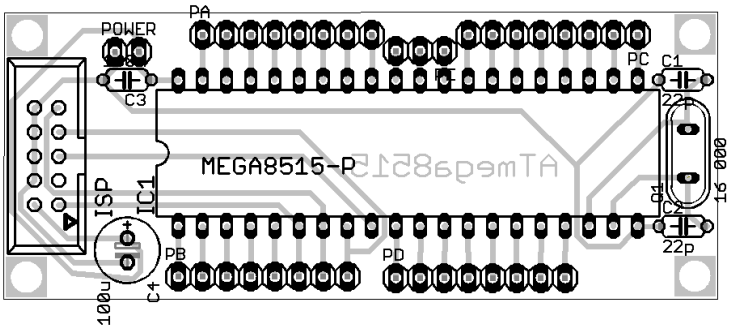
Szczegóły budowy układu sterującego przedstawiają poniższe schematy.



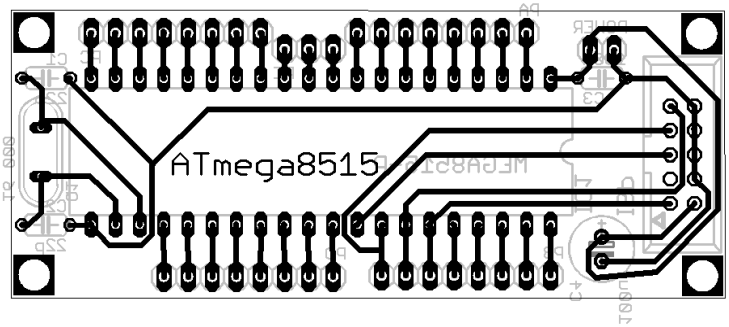
Rys. 4.3.3-1. Schemat ideowy modułu.



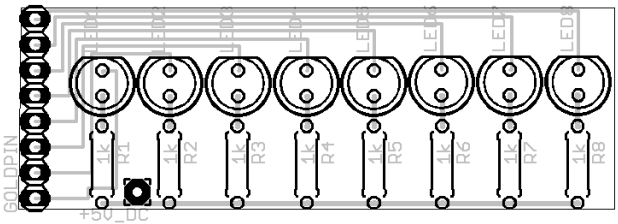
Rys. 4.3.3-2. Schemat ideowy wyświetlacza diodowego.



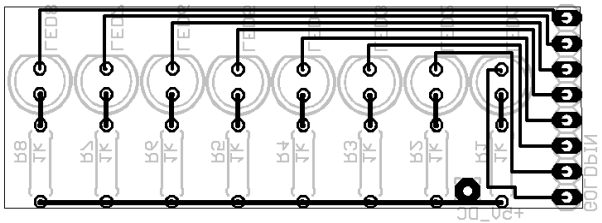
Rys. 4.3.3-3. Płytka drukowana modułu – strona układów.



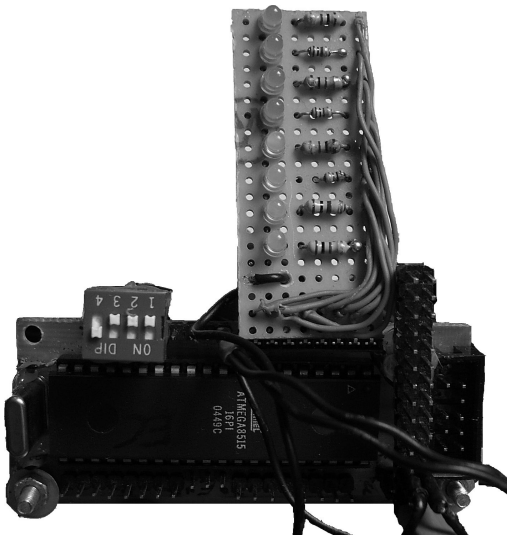
Rys. 4.3.3-4. Płytka drukowana modułu – strona ścieżek.



Rys. 4.3.3-5. Płytka drukowana wyświetlacza diodowego – strona układów.



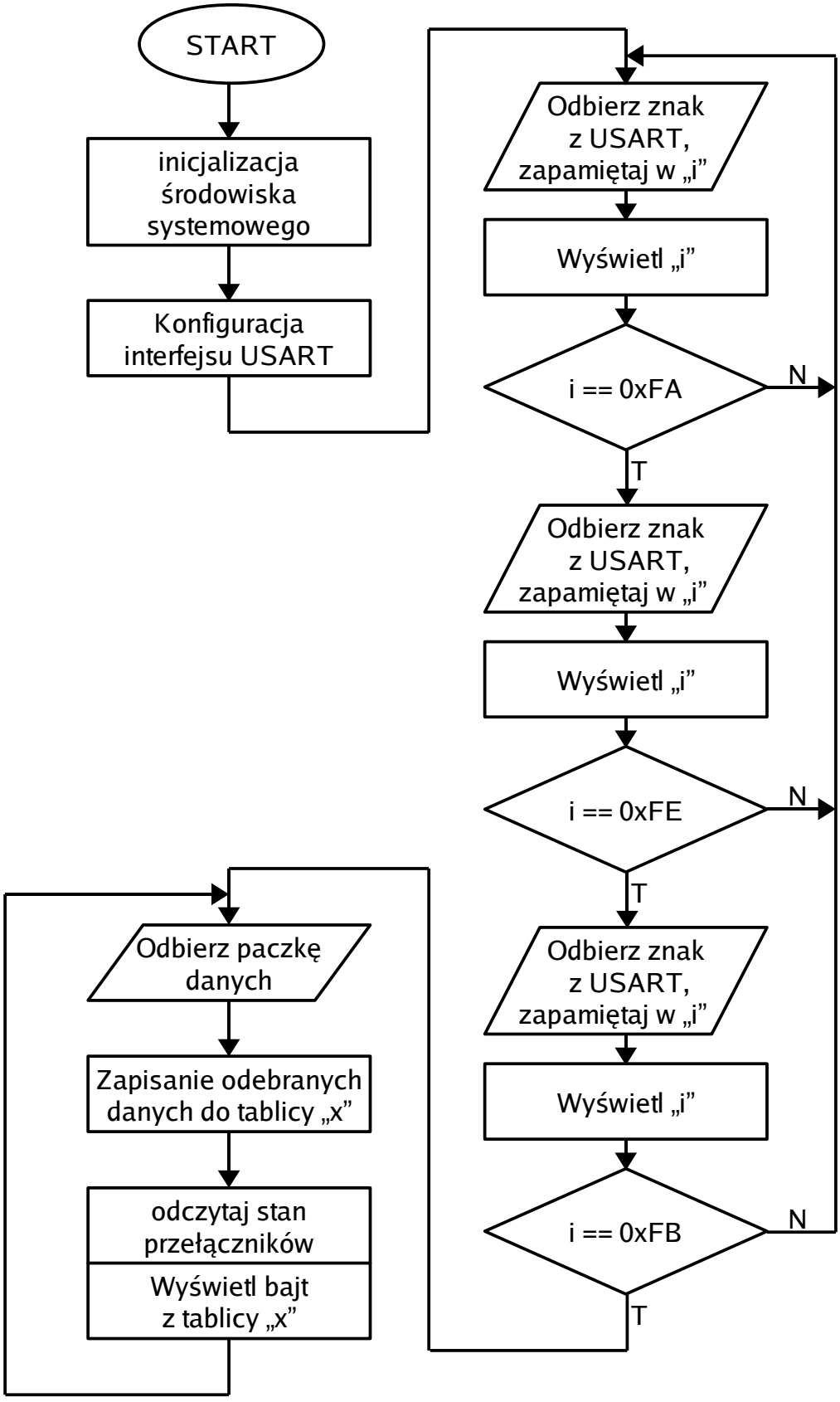
Rys. 4.3.3-6. Płytka drukowana wyświetlacza diodowego – strona układów.



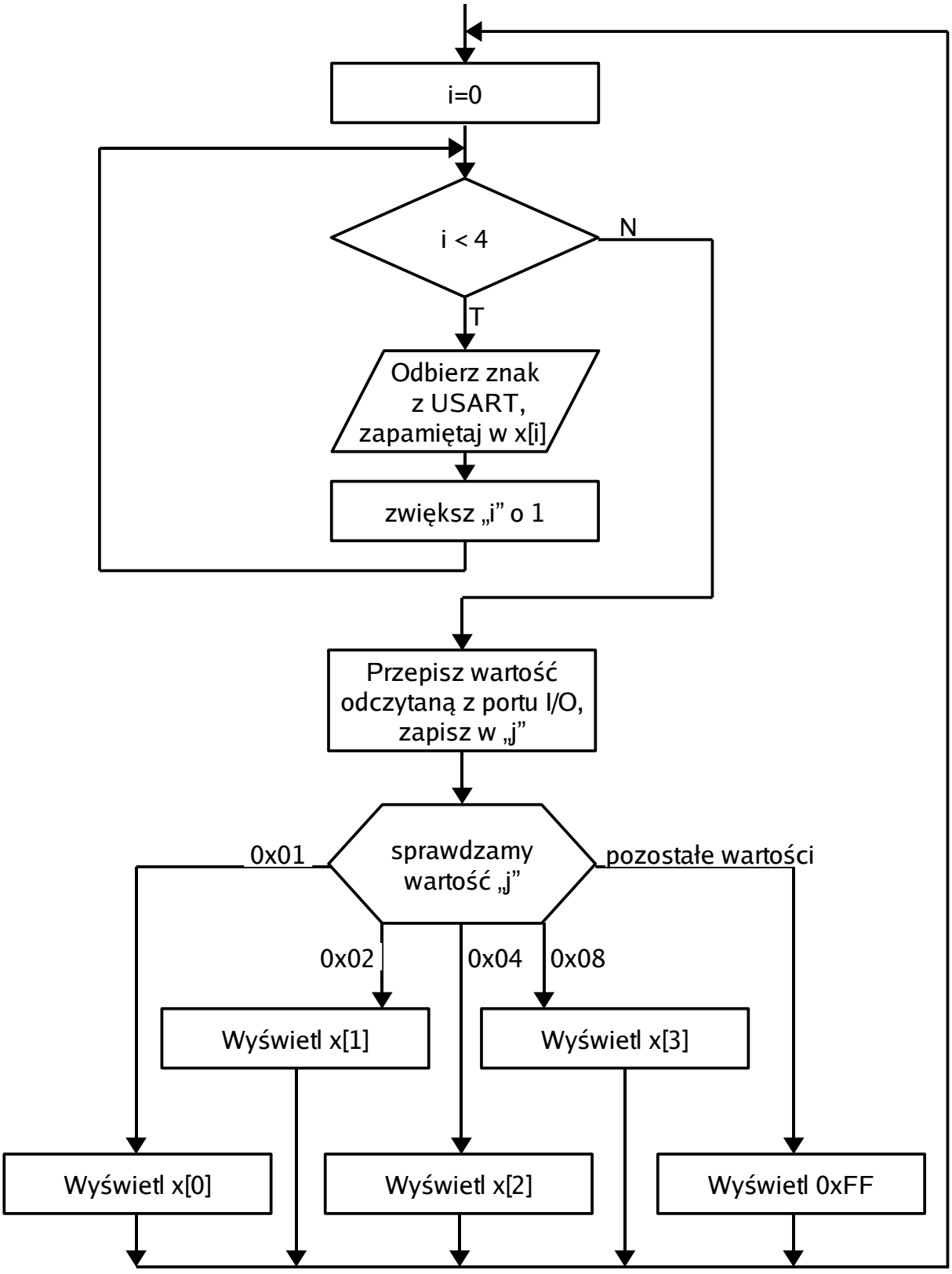
Rys. 4.3.3-7. Zdjęcie zmontowanego modułu.

4.3.4. Opis działania modułu

Oprogramowanie modułu pozwala mu pracować w dwóch trybach. Pierwszy tryb polega na cyklicznym odczycie pojedynczej ramki danych z szeregowej linii transmisyjnej oraz wyświetleniu interpretacji bitowej odebranych danych na wyświetlaczu diodowym. Tryb ten został zaimplementowany ze względu na konieczność zobrazowania danych wysyłanych przez sterownik analogowy w momencie kalibracji. W tym trybie pracy niemożna jednak zrealizować selektywnego wyświetlania pojedynczego bajtu z pakietu danych ponieważ odebranie każdej następnej ramki danych spowodowało by zmianę wskazywanej na wyświetlaczu wartości. Wyświetlenie kolejnych różnych bajtów w bardzo krótkim czasie powoduje nakładanie się wyników na wyświetlaczu uniemożliwiając prawidłowy odczyt. Drugi tryb pracy modułu polega na odbieraniu pakietu danych przedstawionego na rysunku 4.2.4-2, który następnie jest umieszczany w czteroelementowej tablicy z której można wybrać jeden bajt do wyświetlenia. Wybór wyświetlanej części pakietu danych następuje poprzez przesunięcie odpowiedniego przełącznika na pozycje „ON”. Zmiana trybu pracy następuje automatycznie po rozpoznaniu w strumieniu odbieranych danych, ciągu stanowiącego pakiet synchronizujący (Rys. 4.2.4-1). Układ po włączeniu zasilania inicjalizuje zmienne odpowiadające za prawidłową obsługę interfejsu USART, wyświetlacza diodowego oraz przełączników. Następnie Moduł przechodzi w pierwszy z opisanych powyżej trybów pracy, jednocześnie cały czas sprawdzając czy nie nadszedł pakiet synchronizujący.



Rys. 4.3.4-1. Algorytm programu mikrokontrolera.



Rys. 4.3.4-1. Algorytm odbierania i wyświetlania danych.

Ustawienie trybu pracy portów układu.

```
DDRB=0xFF;        // PORTB pracuje jako wyjście..
PORTB=0xFF;       // Wocząkowy stan jest ustawiony na stan wysoki.
DDRD=0x00;        // PORTD pracuje jako wejście.
PORTD=0xFF;       // zostają włączone rezystory podciągające linie portu do zasilania.

uint8_t x[4],i,j;
```

Pętla oczekiwania na synchronizacje, jednocześnie wyświetlane są odebrane informacje.

```
do
{
  x[0]=USART_receive();
  PORTB=~x[0];
  if(x[0]==0xFA)
  {
    x[1]=USART_receive();
    if(x[1]==0xFE)
    {
      x[2]=USART_receive();
      if(x[2]==0xFB)
        x[3]=0xFF;
    }
  }
}
while(x[3]!=0xFF);
```

Główna pętla programu

```
while(1)
{
```

Odebranie pakietu danych i zapisanie go w tablicy

```
  for(i=0;i<4;i++)
    x[i]=USART_receive();
```

Odczytanie stanów przełączników

```
  j=PIND;
  j=~j;
  j>>=4;
```

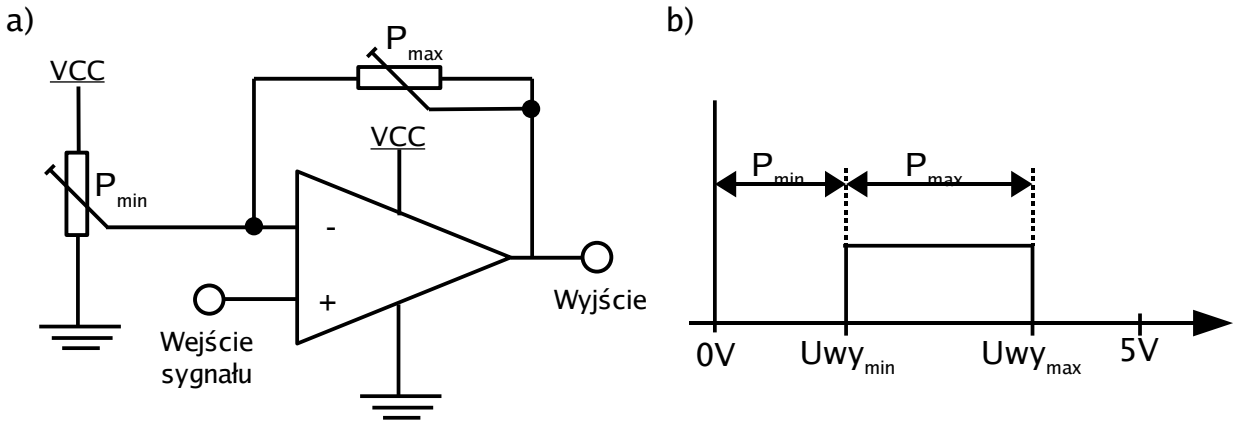
Instrukcja która w zależności od włączonego przełącznika wyświetla pojedynczy bajt danych

```
  switch(j)
  {
    case 0x01: PORTB=~x[0]; break;
    case 0x02: PORTB=~x[1]; break;
    case 0x04: PORTB=~x[2]; break;
    case 0x08: PORTB=~x[3]; break;
    default: PORTB=0x00;
  }
}
```

5. Analogowy kontroler

5.1 Charakterystyka urządzenia

Najważniejsza częścią systemu sterowania jest analogowy joystick, to on jest interfejsem użytkownika, który pozwala na precyzyjne sterowanie pojazdem. Mikrokontroler wbudowany w joystick przetwarza odczytane wartości wychylenia drążka joysticka oraz stany przycisków klawiatury matrycowej oraz następnie przetwarza je na sygnały zrozumiałe dla pozostałych modułów systemu sterowania. Budowa urządzenia wymusiła połączenie w jednym układzie elektronicznym, elementów techniki cyfrowej oraz analogowej. Główny element części cyfrowej stanowi mikrokontroler Atmel Atmega8, który pobiera informacje z układów analogowych za pomocą wbudowanego przetwornika analogowo-cyfrowego (ADC). Cześć analogową tworzą potencjometry mierzące wychylenie drążka oraz wzmacniacze operacyjne podnoszące amplitudę sygnału do poziomu niezbędnego do przetworzenia go z odpowiednią dokładnością.



Rys. 5.1.-1. Schemat i działanie układu dostosowania sygnały
a) Budowa układu pojedynczego wzmacniacza.
b) Wpływ regulacji na sygnał wyjściowy układu.

Ze względu na poprawną prace układu konieczna jest kalibracja joysticka. Kalibracji podlega zarówno część analogowa układu jak i cyfrowa. Kalibracja części analogowej polega na ustawieniu odpowiednich wartości na potencjometrach odpowiadających za parametry pracy wzmacniaczy operacyjnych. Każdorazowa regulacja nie jest konieczna w przypadku zasilania joysticka napięciem o stałej znanej wartości. Zmiana wartości napięcia zasilającego o wartość ok. 0,2V skutkuje znacznie mniejszą dokładnością sterowania. Problem różnic w napięciu zasilania występuje podczas zmiany źródła zasilania urządzenia np. w przypadku podłączenia joysticka bezpośrednio do pojazdu podczas gdy wcześniej działał zasilany z portu USB komputera. Na rysunku 5.1-1 przedstawiono budowę układu wzmacniacza, który stanowi część analogowego układu kalibrującego. Kalibracja układu polega na odpowiednim ustawieniu potencjometrów P_{min} oraz P_{max}, dla kolejnych wzmacniaczy, tak by wartość U_{min} była jak najbliższa poziomowi 0V, a wartość U_{max} była jak najbliżej poziomowi napięcia zasilania (ok. 5V). Kalibracja części cyfrowej jest wykonywana każdorazowo po uruchomieniu urządzenia. Polega ona na pomiarze wartości sygnałów docierających do mikrokontrolera, kiedy drążek jest w pozycjach, centralnej oraz skrajnego wychylenia w każdą stronę. Następnie mierzona jest wartość skrajnych pozycji przepustnicy. Dane otrzymane w tym procesie służą jako parametry do późniejszego przeskalowania odczytywanych z części analogowej wartości, które są przesyłane pozostałym modułom systemu sterowania. Tak przygotowane dane gwarantują unikniecie przekłamań związanych ze zużyciem mechanicznych elementów urządzenia.

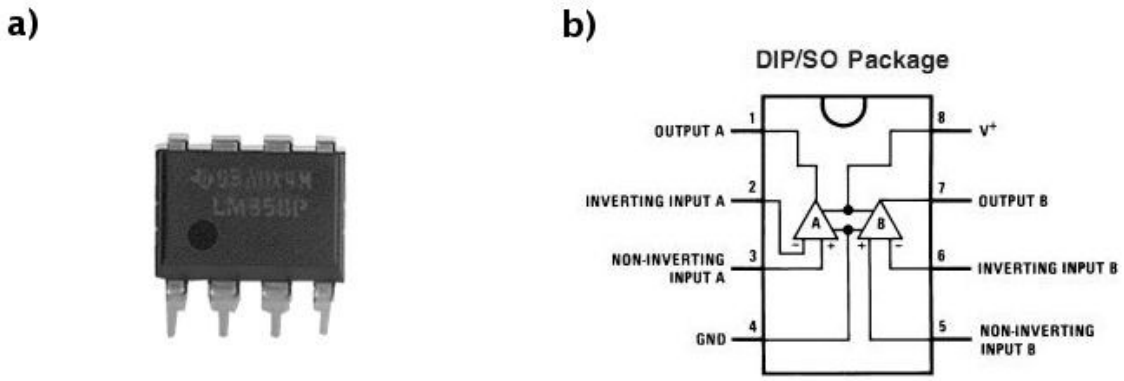
Joystick dodatkowo wyposażony jest w gniazdo USB podłączone do mikrokontrolera w sposób umożliwiający programowe zrealizowanie komunikacji przez port USB. Rozwiązanie to umożliwia podłączenie alternatywnego zasilania układu. Dodatkowo projekt urządzenia pozwala na implementację możliwości komunikacji przez wspomniany port, bez konieczności wykonania nowej wersji układu. Modernizacja możliwości urządzenia sprowadza się jedynie do napisania nowego oprogramowania.

5.2 Charakterystyka zastosowanych układów

Do budowy części analogowej zastosowano uniwersalne wzmacniacze operacyjne LM358, charakteryzujące się pojedynczym napięciem zasilania i małym poborem prądu.

Lista najbardziej istotnych parametrów układu LM358:

- Pojedyncze napięcie zasilania od 3 do 30V.
- Pobór prądu na poziomie ok. 7mA.
- zawiera dwa wzmacniacze operacyjne w jednej obudowie.
- Prąd zasilający niezależny od napięcia zasilania.

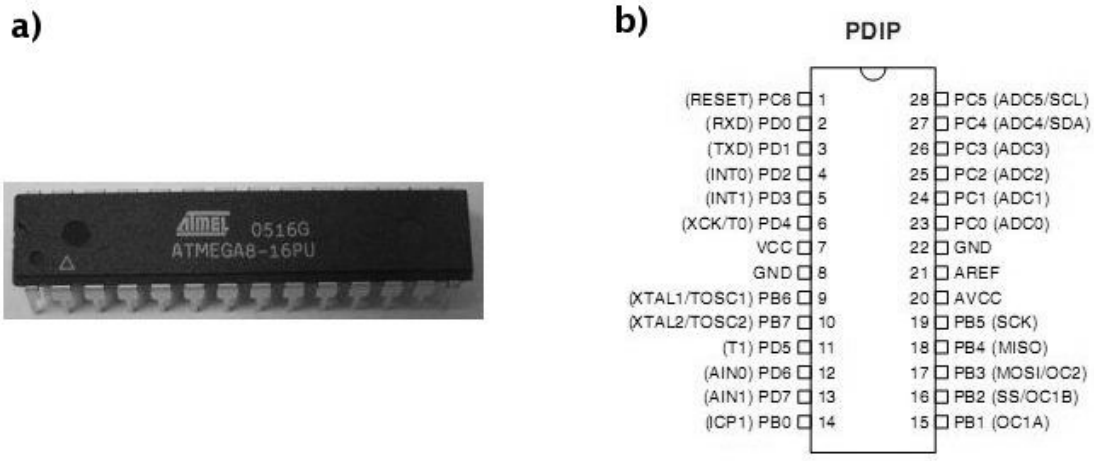


Rys. 5.2.-1. Układ LM358 [13]
a) Zdjęcie.
b) Rozkład wyprowadzeń.

Cyfrowa część układu została zbudowana na bazie mikrokontrolera Atmega8, który charakteryzuje się dużą wydajnością obliczeniową konieczną do szybkiego wykonywania operacji na liczbach zmiennoprzecinkowych. Dodatkowo posiada odpowiednią ilość zintegrowanych w układzie interfejsów peryferyjnych, z których największe znaczenie ma przetwornik analogowo-cyfrowy, wymagany do pracy z układami analogowymi oraz układ USART zapewniający komunikacje z pozostałymi modułami.

Lista najważniejszych parametrów mikrokontrolera przedstawia się w następujący sposób:

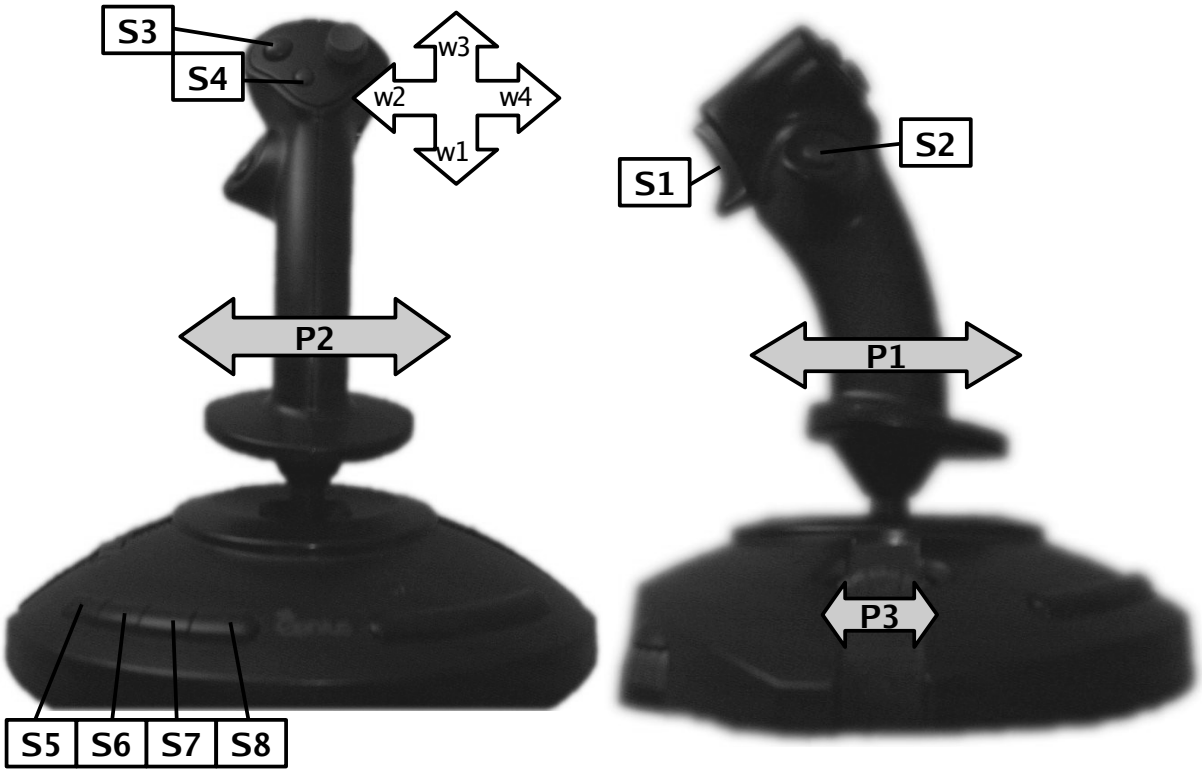
- Wykonanie w technologii CMOS
- Architektura 8-bitowa
- 130 instrukcji procesora – większość zajmuje jeden cykl zegarowy
- 32 jedno-bajtowe rejestry robocze
- Moc obliczeniowa do 1 MIPS na każdy 1 MHz
- 8KB dostępnej pamięci programu typu FLASH
- 512B dostępnej pamięci danych typu EEPROM
- 1kB dostępnej pamięci SRAM
- Peryferia
  - dwa 8-bitowe liczniki/układy czasowy
  - pojedynczy 16-bitowy licznik/układ czasowy
  - 3 kanały PWM
  - USART
  - SPI zapewniający możliwość programowania ISP
- Obsługa wewnętrznych i zewnętrznych źródeł przerwań
- Wewnętrzny generator sygnału zegarowego
- Napięcie zasilania od 4,5V do 5,5V
- Taktowanie zegarem do 16MHz



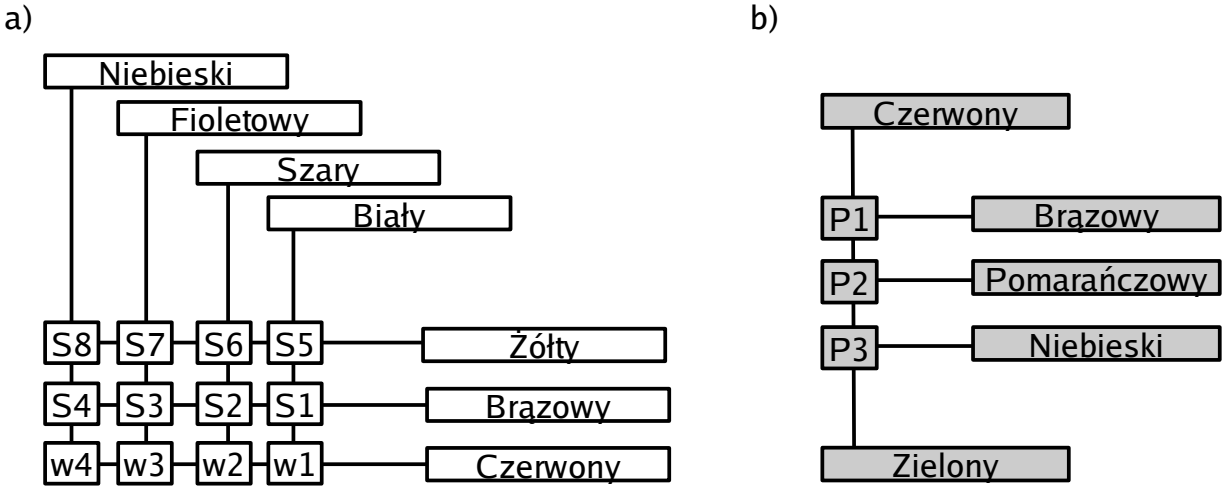
Rys. 5.2.-2. Mikrokontroler Atmega8 [12]
a) Zdjęcie.
b) Rozkład wyprowadzeń.

5.3. Opis budowy urządzenia

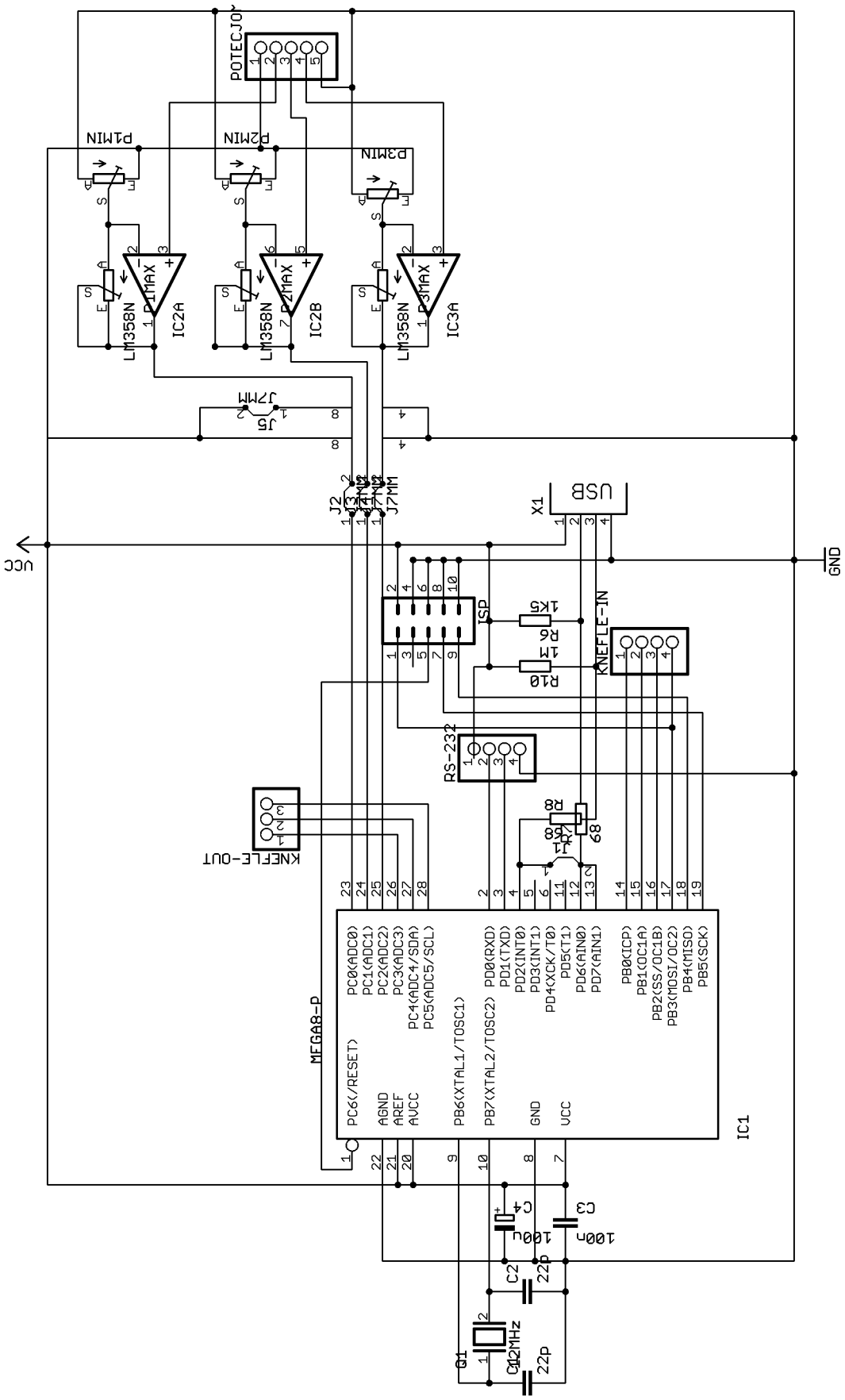
Szczegóły budowy urządzenia przedstawiają poniższe schematy.



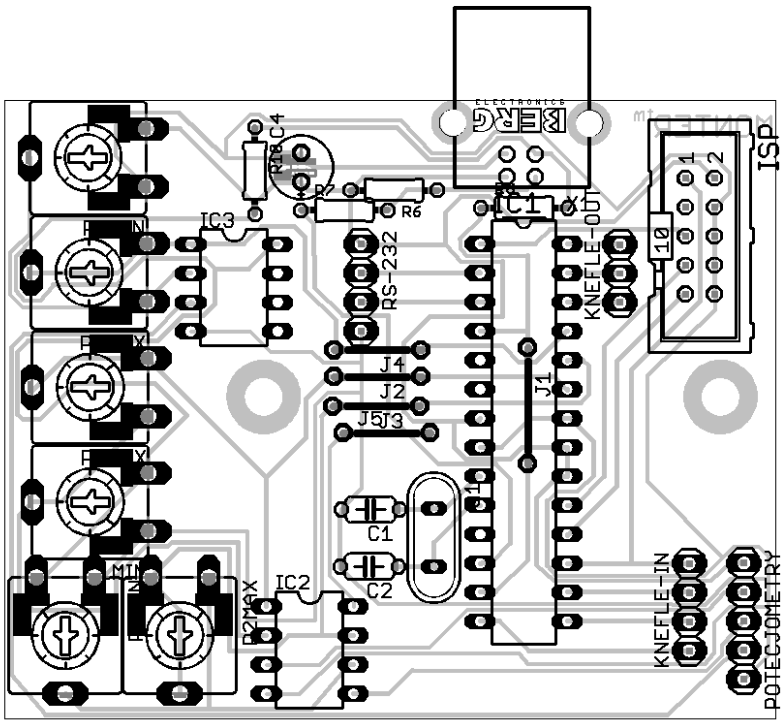
Rys. 5.3-1. Joystick- rozkład potencjometrów (Pn) i przycisków(Sn,wn).



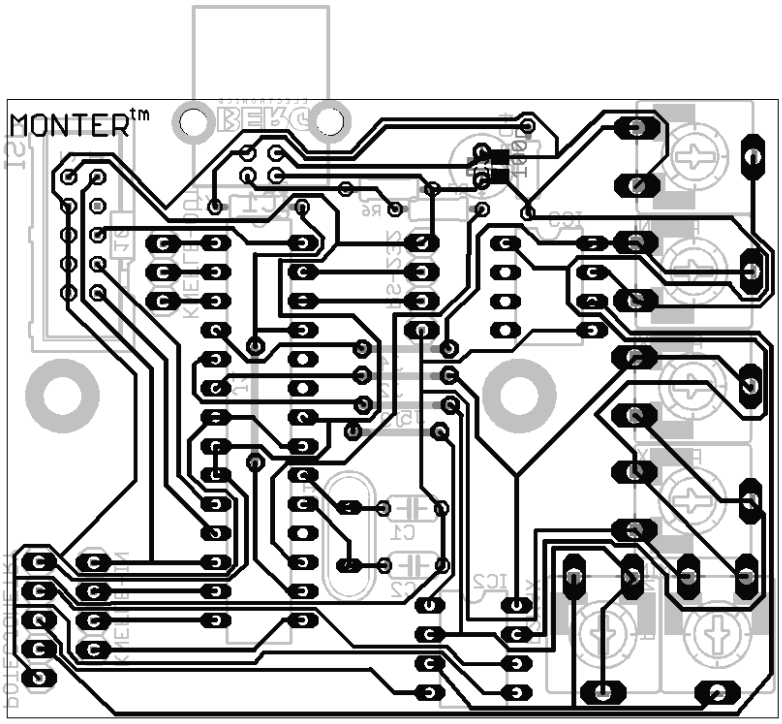
Rys. 5.3-2. Schemat podłączeń – kolory przewodów.
a) Klawiatury matrycowej.
b) Układu potencjometrów.



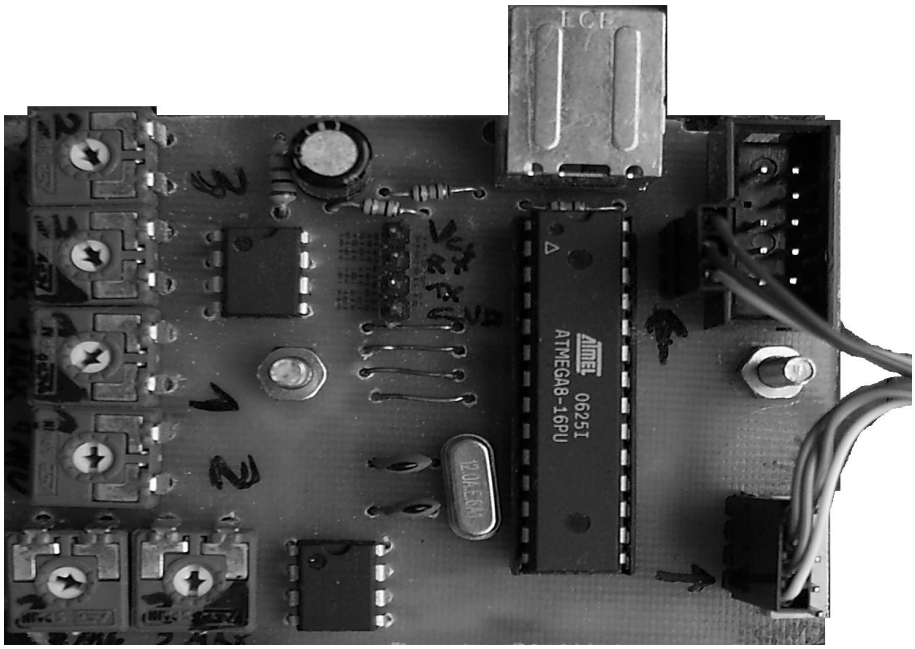
Rys. 5.3-3. Schemat ideowy urządzenia.



Rys. 5.3-4. Płytka drukowana – strona układów.



Rys. 5.3-5. Płytka drukowana – strona ścieżek.



Rys. 5.3-6. Zdjęcie zmontowanego układu.

5.4. Opis działania urządzenia

Pierwszą czynnością jaką wykonuje urządzenie zaraz po włączeniu i inicjalizacji zmiennych systemowych odpowiadających za poprawną prace zintegrowanych układów peryferyjnych, jest przejście do procedury kalibracji części cyfrowej urządzenia. W tym momencie należy też wykonać regulacje pracy układów analogowych, o ile zajdzie takowa potrzeba. Regulacja wymaga ustawienia odpowiednich poziomów napięć na wyjściach układów dostosowujących sygnały analogowe. Do prawidłowego dostrojenia układu można wykorzystać woltomierz, ale w tym przypadku konieczne byłoby podłączenie go do odpowiednich wyjść wzmacniaczy operacyjnych. Tego typu operacja jest dość kłopotliwa ze względu na fakt, iż w projekcie nie uwzględniono specjalnych podłączeń dla woltomierza. W projekcie przewidziano możliwość elektroniczego odczytu poziomów napięć na wyjściach układów dopasowujących poziom sygnałów, tak więc opracowano procedurę kalibracji w sposób pozwalający na dostęp do nieprzetworzonych danych odczytanych z przetwornika analogowo-cyfrowego. Są one dostępne w czasie rzeczywistym na zewnątrz układu, ponieważ dane te są wysyłane na bieżąco portem szeregowym, a następnie wyświetlane za pomocą modułu linijki diodowej.

Procedura kalibracji joysticka jest najbardziej skomplikowaną czynnością podczas obsługi

systemu. Aby poprawnie skalibrować urządzenie należy po włączeniu urządzenia wykonać procedurę według następującego algorytmu.

1. Przechylić drążek maksymalnie do tyłu.
2. Dokonać regulacji na potencjometrze $P1_{min}$ kontrolując wartość na linijce diodowej
3. Przechylić drążek maksymalnie do przodu
4. Dokonać regulacji na potencjometrze $P1_{max}$ kontrolując wartość na linijce diodowej
5. Powtórzyć czynności 1-4 do momentu uzyskania optymalnych wartości.
6. Ustawić drążek kolejno w pozycjach maksymalnego wychylenia w tył, centralnej, maksymalnego wychylenia w przód, zatwierdzając moment osiągnięcia każdej pozycji przyciśnięciem przycisku S2.
7. Analogicznie z punktami 1-6 wykonać czynności dla odpowiednio kierunków w prawo, w lewo oraz potencjometrów $P2_{min}$, $P2_{max}$.
8. Analogicznie z punktami 1-6 wykonać czynności dla odpowiednio przepustnicy ustawionej w pozycji maksymalnej tył, maksymalnej przód oraz potencjometrów $P3_{min}$, $P3_{max}$. Pomijając zatwierdzenie pozycji centralnej.

Po zakończeniu procesu kalibracji automatycznie wysyłany jest pakiet synchronizujący zgodny z pakietem opisanym na rysunku 4.2.4-1. Następnie cyklicznie odczytywane są wartości z przetwornika analogowo-cyfrowego świadczące o stanie wychylenia drążka, pozycji przepustnicy i stanie przycisków. Wartości te są odpowiednio przetwarzane oraz przesyłane w pakiecie danych przedstawionym na rysunku 5.4-1.

Pierwszy bajt	**Drugi bajt**	**Trzeci bajt**	**Czwarty bajt**
Stopień wychylenia przód/tył w skali 0-200	Stopień wychylenia prawo/lewo w skali 0-200	Pozycja przepustnicy w zakresie 0-200	Stan przycisków S8-S1 bit=1 – przycisk wciśniety

Rys. 5.4-1. Paczka danych.

Procedura wysyłania pojedynczego bajtu danych.

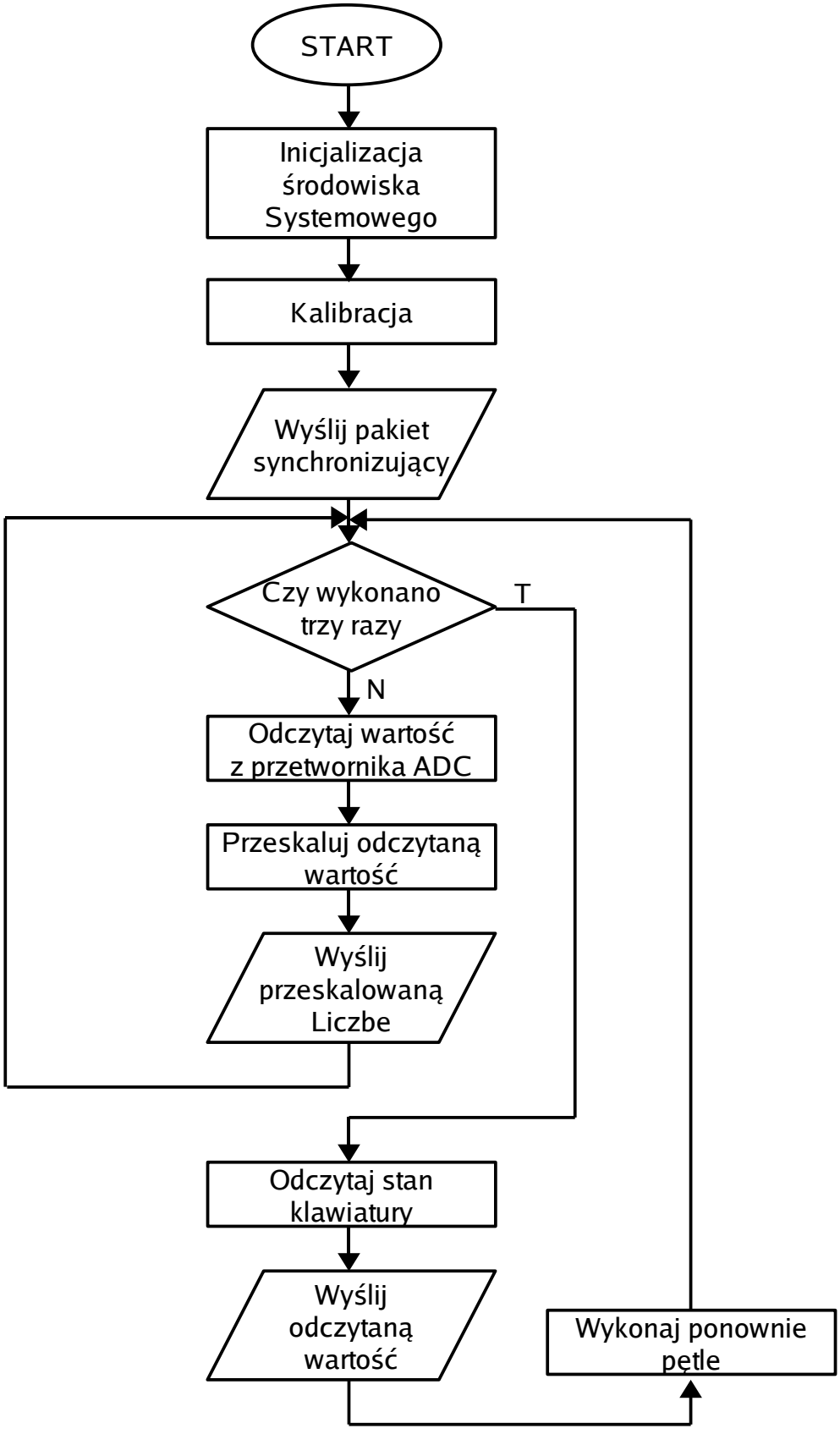
```
void USART_transmit(uint8_t data)
{
  UDR = data;
  while( !(UCSRA & 0x40) );
}
```

Odczytanie wartości z przetwornika analogowo cyfrowego.

```
uint8_t ADC_read(uint8_t mux)
{
  ADMUX&=0xF0;
  ADMUX|=mux;             // Wybieramy numer wejścia analogowego z którego układ
                          // ADC będzie odczytywać dane.
  ADCSRA|=0x40;           // Włączenie procesu konwersji (A->C)
  while(ADCSRA&0x40);     // Czkanie do chwili zakończenia zakończenia.
  return ADCH;            // Zwrócenie odczytanej z ADC wartości.
}
```

Odczytanie stanu klawiatury

```
uint8_t KEY_read(uint8_t select)
{
  uint8_t x;
  if(!select)         // odczytaj stan przycisków S1-S8
  {
    PORTC=0xF7;
    _delay_ms(10);
    x=PINB;
    x<<=4;
    PORTC=0xEF;
    _delay_ms(10);
    x|=PINB;
  }
  else                // odczytaj stan przycisków w1-w4
  {
    PORTC=0xDF;
    _delay_ms(10);
    x=PINB;
    x|=0xF0;
  }
  return ~x;
}
```



Rys. 5.4-2. Algorytm programu mikrokontrolera.

Włączenie oraz skonfigurowanie interfejsu komunikacji szeregowej.

```
void USART_init(void)
{
  UBRRH = 0x00;       // Starszy bajt ustawiający prędkość transmisji.
  UBRRL = 0x4D;       // Młodszy bajt ustawiający prędkość transmisji
  UCSRB = 0x18;       // Włączenie nadajnika oraz odbiornika.
  UCSRC = 0x8E;       // Ustawienie trybu pracy na asynchroniczny, ustawienie formatu
                      // ramki na 8 bitów danych, bez parzystości oraz dwa bity stopu.
}

void ADC_init(void)
{
  ADMUX=0x20;         // Ustawienie źródła napięcia odniesienia.
  ADCSRA=0x87;        // Włączenie przetwornika ADC, ustawienie zegara taktującego układ
                      // ADC.
}
```

Konfiguracja uniwersalnych portów do obsługi klawiatury matrycowej.

```
void KEY_init(void)
{
  DDRC=0x38;    // Ustawienie poszczególnych linii portu jako wyjście.
  PORTC=0x38;   // Ustawienie początkowego stanu wyjść na wartość wysoka.
  DDRB=0xF0;    // Ustawienie poszczególnych linii portów jako wejście
  PORTB=0x0F;   // Włączenie rezystora podciągającego do zasilania
}
```

Główna funkcja programu

```
int main(void)
{
  USART_init();
  ADC_init();
  KEY_init();

  uint8_t mux;

  calibration();
```

Wysłanie pakietu synchronizującego.

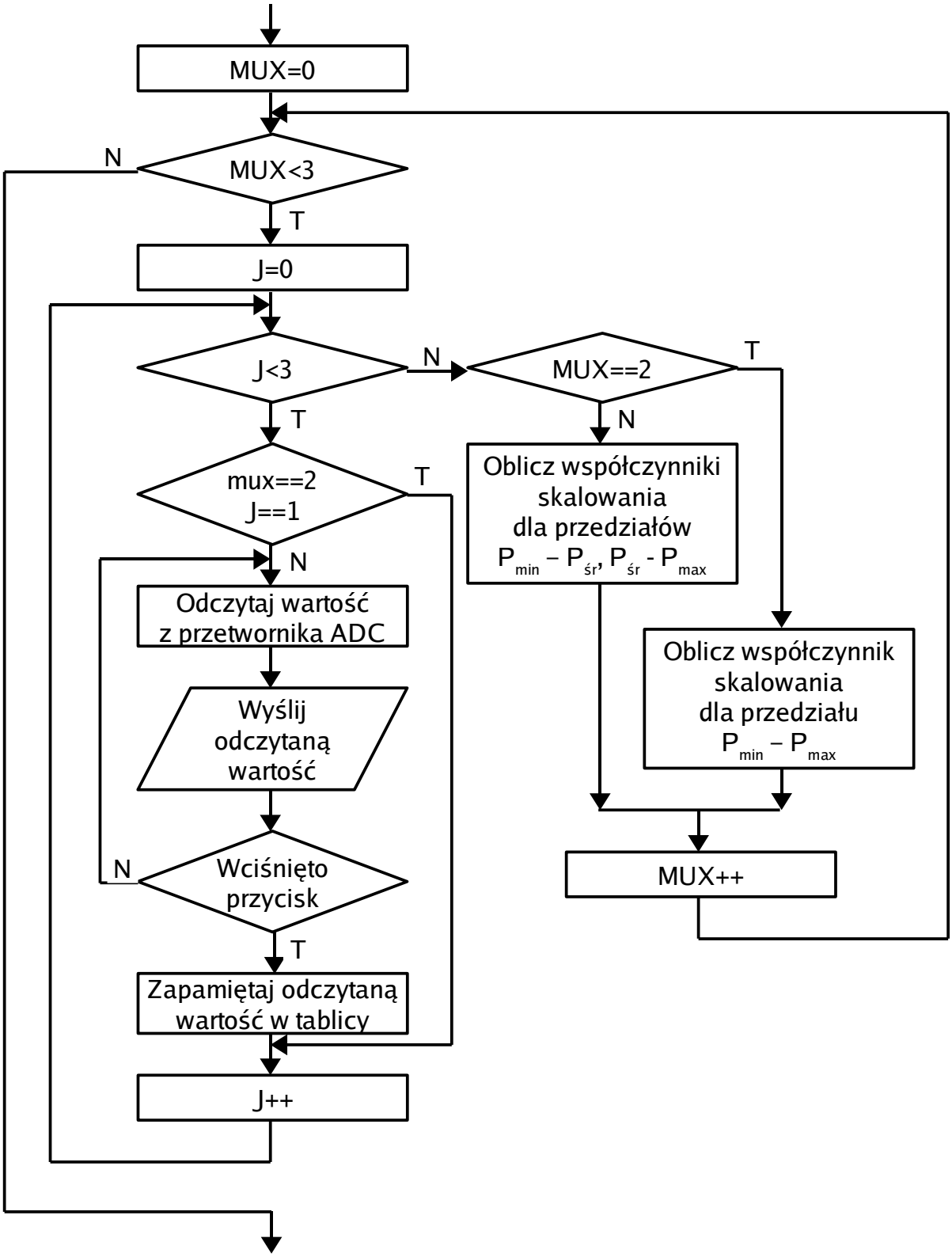
```
  USART_transmit(0xFA);
  _delay_ms(15);
  USART_transmit(0xFE);
  _delay_ms(15);
  USART_transmit(0xFB);
  _delay_ms(15);
```

Główna pętla programu.

```
  while(1)
  {
    for(mux=0;mux<3;mux++)
    {
      USART_transmit(TRIM_set(mux,ADC_read(mux)));
      _delay_ms(10);
    }
    USART_transmit(KEY_read(0));
    _delay_ms(10);
  }
}
```

Przeskalowywanie odczytanej wartości wartości z potencjometru.

```
uint8_t TRIM_set(uint8_t mux,uint8_t adc)
{
  uint8_t tr_set;
  if(mux!=2)
  {
    if(adc<(TRYM[mux][1]-10))
      tr_set=res[mux][0]*(adc-TRYM[mux][0]);
    else
    {
      if(adc<=(TRYM[mux][1]+10))
        tr_set=0x64;
      else
        tr_set=0x64+res[mux][1]*(adc-TRYM[mux][1]);
    }
  }
  else
    tr_set=res[mux][0]*(adc-TRYM[mux][0]);
  return tr_set;
}
```



Rys. 5.4-3. Algorytm kalibracji urządzenia.

Proccedura kalibracji.

```
void calibration(void)
{
  uint8_t mux,j,x,tmp;
  for(mux=0;mux<3;mux++)
  {
    for(j=0;j<3;j++)
    {
      if(mux==2 && j==1) continue;
      do
      {
        tmp=ADC_read(mux);
        USART_transmit(tmp);
        x=KEY_read(0);
      }
      while((x&0x01)==0);
      do
      {
        x=KEY_read(0);
      }
      while((x&0x01)==1);
      TRYM[mux][j]=tmp;
    }
    if(mux!=2)
    {
      res[mux][0]=(float)((float)100/(TRYM[mux][1]-TRYM[mux][0]));
      res[mux][1]=(float)((float)100/(TRYM[mux][2]-TRYM[mux][1]));
    }
    else
      res[mux][0]=(float)((float)200/(TRYM[mux][2]-TRYM[mux][0]));
  }
}
```

6. Platforma rozwojowa ARM

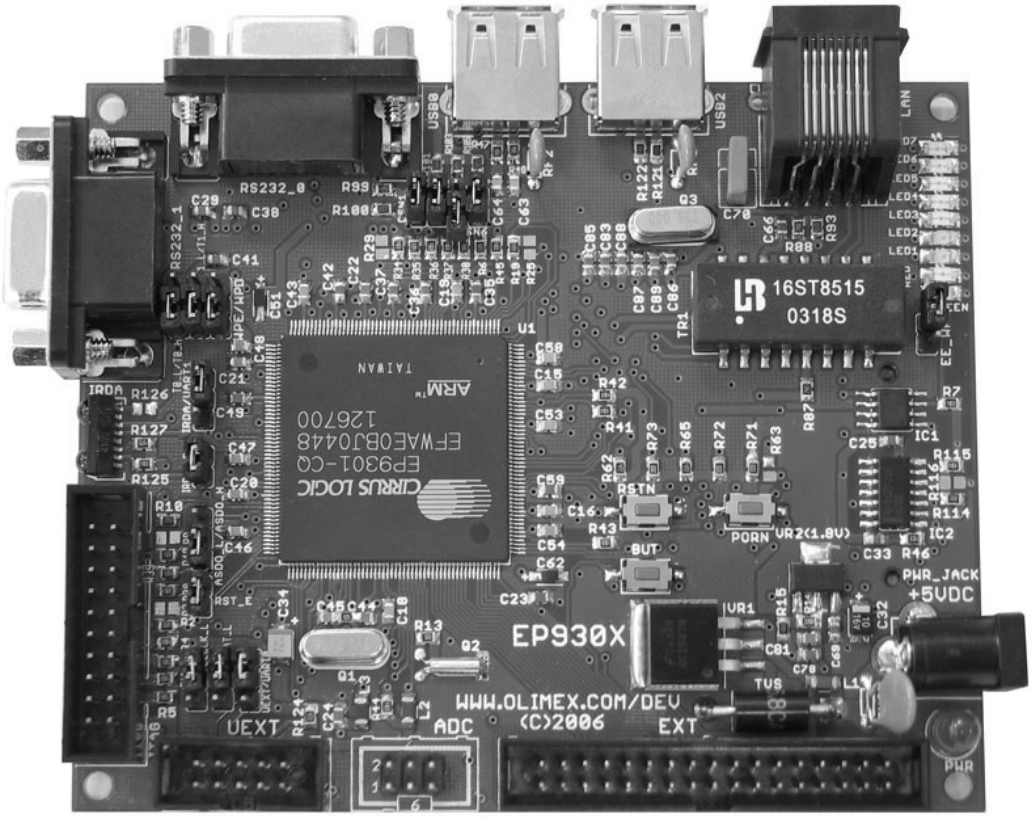
W zamyśle budowa systemu sterowania pojazdem miała opierać się na zastosowaniu gotowego modułu rozwojowego z procesorem typu ARM, który miał spełniać rolę zaawansowanego komputera pokładowego. Decyzja o wyborze architektury ARM podyktowana była bardzo dużymi możliwościami i uniwersalnością obecnie produkowanych modeli mikroprocesorów. Atutem tego rozwiązania jest także bardzo mały pobór prądu wynikający z energooszczędnej architektury. Energooszczędność układu była jednym z kluczowych parametrów tego rozwiązania, ponieważ układ docelowo miał być zamontowany na pojeździe zasilanym z akumulatorów. Po zapoznaniu się z ofertami dostępnych na rynku modułów rozwojowych opartych na procesorze typu ARM, postanowiono zastosować, w budowanym systemie sterowania, model płytki rozwojowej CS-E9302. Układ został zaprojektowany oraz zbudowany przez firme Olimex (http://olimex.com/dev/), jego budowa została oparta na mikrokontrolerze CirrusLogic EP9302.

Najważniejsze parametry mikrokontrolera EP9302 [14].

- Budowa oparta na rdzeniu ARM920T 200Mhz
- 16+16KB pamięci cache instrukcji i danych
- Interfejs obsługi pamięci SDRAM, SRAM oraz FLASH na magistrali 66MHz
- Interfejsy komunikacji
  - ethernet 100Mb
  - dwa porty UART
  - dwa porty host USB
  - IrDA
  - SPI
- 12-bitowy przetwornik analogowo-cyfrowy
- Podwójny 16-bitowy oraz pojedynczy 32-bitowy układ czasowy
- Boot ROM
- Kontroler przerwań oraz 12-kanałowy układ DMA

Właściwości płytki rozwojowej CS-E9302 [14].

- Mikrokontroler EP9302
- Zewnętrzna pamięć SDRAM 32MB (16MB x 16bit) 7.5 ns /133Mhz
- zewnętrzna pamięć Flash 16MB (8MB x 16bit) 80 ns
- ETHERNET 10/100 PHY CS8952
- 2 x RS232, 2 x USB, IrDA
- złącze kart SD/MMC
- złącze UEXT do podłączenia dodatkowych modułów
- interfejs JTAG
- złącze ADC z sygnałami analogowymi



Rys. 6-1. Widok modułu rozwojowego z góry.

Przedstawiony moduł może pracować pod kontrolą, pojedynczego programu lub systemu operacyjnego, zapisanego w pamięci FLASH. Praca z pojedynczymi programami znacznie utrudnia wykorzystanie pełnych możliwości sprzętowych układu, gdyż korzystanie w programie ze zintegrowanych układów wiązałoby się z opracowaniem skomplikowanych algorytmów. Natomiast pod kontrolą systemu operacyjnego moduł staje się zaawansowanym urządzeniem

pozwalającym na tworzenie, kompilacje oraz uruchamianie oprogramowania, która ma działać na przedstawionej platformie sprzętowej.

Wykorzystanie możliwości pracy modułu wraz z systemem operacyjnym, wymaga jedynie zaopatrzenia urządzenia w zewnętrzny interfejs użytkownika, który może stanowić pasywny terminal tekstowy połączony z modułem poprzez interfejs RS232. Fabrycznie nowy układ został przystosowany do pracy z systemem operacyjnym Linux lub NetBSD, które zostały zamieszczone, w postaci pakietów źródłowych, na dołączonej płycie cd-rom. Źródła jądra systemu linux dostępne są w dwóch wersjach: 2.4.21 oraz 2.6.8.1, do każdej z nich producent dodał poprawki, które zawierają sterowniki wszystkich urządzeń składających się na budowę modułu. Aby uruchomić system linux-2.6.8.1 należy w pierwszej kolejności skompilować binarny obraz jądra systemu, który następnie należy zapisać w pamięci FLASH układu oraz skonfigurować program bootstrap ładujący oraz uruchamiający system operacyjny.

Proces kompilacji jądra systemu należy wykonać na komputerze z zainstalowanym środowiskiem programistycznym dla architektury ARM. Środowisko to, tak samo jak w przypadku mikrokontrolerów z rodziny AVR, składa się z kompilatora GCC, biblioteki libc oraz pakietu binutils. Wszystkie wymienione elementy znajdują się na załączonej do modułu, płycie cd-rom, w spakowanym archiwum. W celu poprawnego i bezbłędnego działania środowiska programistycznego w systemie linux, należy rozpakować archiwum „arm-linux-gcc-3.4.1.tar.bz2" do systemowego katalogu głównego, następnie do zmiennej PATH należy dodać wpis „/usr/local/arm/3.4.1/bin". Dopiero w tak przygotowanym środowisku można przeprowadzić kompilacje systemu. Kompilacja jest często procesem żmudnym i długotrwałym, ponieważ kompilator musi przetłumaczyć ogromną liczbę linii kodu, składających się na pliki źródłowe. Jednocześnie w większości przypadków trzeba kilkakrotnie przeprowadzać kompilację od początku, gdyż okazuje się, że wygenerowany obraz jądra systemu jest błędnie zbudowany lub nie zawiera wszystkich, koniecznych do poprawnego działania sterowników. Czasami także podczas procesu kompilacji pojawia się nieznany błąd, który powoduje przerwanie procesu kompilacji, wymuszając konieczność ponownej konfiguracji i kompilacji jądra systemu.

Lista kroków jakie należy wykonać przy kompilacji jądra systemu linux.

1. Skopiowanie archiwum zawierającego źródła jądra systemowego, do katalogu tymczasowego (w przykładzie użyto katalogu domowego użytkownika).
   ```
   cp /mnt/cdrom/linux sources/linux-2.6.8.1.tar.bz2 ~/
   ```
2. Przejście do katalogu domowego oraz rozpakowanie archiwum.
   ```
   cd ~/
   ```

```
tar –xvjf linux–2.6.8.1.tar.bz2
```

3. Zainstalowanie łatki systemu oraz przejście do katalogu ze źródłami systemu.

   ```
   patch –p0 < linux–2.6.8.1–cirrus.diff
   cd linux–2.6.8.1
   ```

4. Zmiana wartości zmiennych w skrypcie makefile.

   ```
   nano –w Makefile
   ```

   W pliku należy zmienić następujące pozycje.

   `ARCH ?= $(SUBARCH)` zamienić na `ARCH ?= arm`

   `CROSS_COMPILE ?=` zamienić na

   `CROSS_COMPILE ?= /usr/local/arm/3.4.1/bin/arm–linux–`

5. Wykonanie wstępnej konfiguracji procesu kompilacji.

   ```
   make edb9301_defconfig
   ```

6. Dokonanie wyboru modułów jądra systemu które mają znaleźć się w docelowym obrazie jądra. Po zaznaczeniu/odznaczeniu odpowiednich pól należy zapisać konfiguracje.

   ```
   make menuconfig
   ```

7. Uruchomienie procesu kompilacji

   ```
   make clean
   make zImage
   make modules
   ```

Dokonanie nieprawidłowego wyboru podczas operacji z punktu 6, w większości przypadków generuje błędy podczas kompilacji lub późniejszego uruchamiania systemu. Nie należy odznaczać lub zaznaczać opcji, których działanie nie jest znane.

Kolejnym krokiem do uruchomienia systemu linux na platformie rozwojowej jest przesłanie odpowiednich plików do pamięci FLASH układu oraz uruchomienie systemu. Pierwszym plikiem jaki należy przesłać do modułu jest plik o nazwie „ramdisk.gz", jest to spakowany system plików, który zawiera podstawowe drzewo katalogów oraz programy niezbędne do prawidłowej pracy systemu linux. Plik ramdisk.gz można znaleźć na płycie cd-rom dołączonej do zakupionego układu. Kolejnym plikiem jaki należy przenieść do pamięci do pamięci układu jest wcześniej utworzony obraz jądra systemu. Na wgranie wyżej wymienionych plików pozwala program Redboot, zainstalowany w pamięci urządzenia. Program ten się automatycznie uruchamia zaraz po podłączeniu zasilania do układu. Procedurę przenoszenia plików do układu należy zacząć od uruchomienia układu i podłączenia terminalu tekstowego. Terminal tekstowy standardowo jest podłączany do portu szeregowego układu. Domyślne parametry transmisji szeregowej wynoszą

kolejno: prędkość transmisji – 57600kbps, 8 bitów danych, brak parzystości, jeden bit stopu oraz brak kontroli przepływu. Takie same wartości należy ustawić w podłączanym terminalu. Jako terminal użyto programu komputerowego o nazwie „minicom”. Po uruchomieniu i skonfigurowaniu programu oraz po fizycznym połączeniu układu z komputerem, należy włączyć zasilanie układu, w oknie terminala ukażą się informacje świadczące o uruchomieniu programu redboot. następnie należy wcisnąć ctrl+c aby przejść do wiersza poleceń programu redboot. Na tym etapie należy wykonać następujące instrukcje:

1. Wgrać obraz systemu plików do pamięci układu, operacje tę wykonuje się poleceniem
   `load –r –v –b <base_adress> –m ymodem`
   jednocześnie w oknie terminala należy włączyć opcje wysyłania pliku ramdisk.gz.
2. Wgrać obraz jądra systemu w sposób analogiczny do punktu 1.
3. Uruchomić system operacyjny poleceniem
   `exec –r <base_adress> –c ''console=ttyAM root=/dev/ram''`

Jeżeli wszystkie wcześniej wymienione czynności zostały wykonane poprawnie system powinien się prawidłowo uruchomić, dając użytkownikowi do dyspozycji użytkownikowi działający system operacyjny linux [16].

Mimo z pozoru prostej procedury uruchamiania układu, zadanie okazało się dużym wyzwaniem, głównie ze względu na konieczność zrozumienia na dość szczegółowym poziomie dokumentacji technicznej dołączonej przez producenta. Dokładne zrozumienie szczegółów implementacyjnych jest konieczne ze względu na znaczne różnice w budowie systemu mikroprocesorowego i standardowego komputera PC, które w głównej mierze komplikują oraz zwielokrotniają proces kompilacji i uruchamiania systemu operacyjnego. Jednocześnie potrzebna jest szczegółowa wiedza na temat systemu linux, dotycząca zarówno budowy systemu od podstaw, jak i późniejszej obsługi urządzeń.

Proces uruchamiania i konfiguracji systemu operacyjnego okazał się być na tyle pracochłonny, że zaowocował on jedynie zapewnieniem podstawowej funkcjonalności modułu rozwojowego ARM. W chwili obecnej układ jest wyposażony w podstawową instalacje systemu linux opartego na jądrze w wersji 2.6.8.1, która pod względem funkcjonalności odpowiada dystrybucjom systemu linux typu LiveCD, dostępnych na platformie komputera PC. System operacyjny tego typu charakteryzuje się niemożnością przywrócenia zmienionej konfiguracji systemu, po odłączeniu zasilania. Taka wada systemu jest spowodowana faktem iż, system plików stanowi wydzielona część nietrwałej pamięci RAM, której zawartość ulega skasowaniu podczas braku zasilania. Aby wyeliminować tego typu problem należałoby umieścić system plików

na zewnętrznej karcie pamięci lub przenośnej pamięci USB. Jednocześnie komunikacja z układem rozwojowym odbywa się jedynie poprzez port szeregowy, który zapewnia możliwość obsługi systemu poprzez terminal tekstowy.

Tak skonfigurowanego modułu nie udało się zintegrować z pozostałymi elementami systemu sterowania, główną przyczyną takiego stanu rzeczy była pracochłonność procesu kompilacji i konfiguracji systemu operacyjnego, połączona z ograniczeniami czasowymi nałożonymi na projekt. Sprawiło to, iż system sterowania w aktualnej wersji stanowią moduły: sterowania układami wykonawczymi podwozia, linijki diodowej oraz moduł analogowego kontrolera. Moduły te komunikują się bezpośrednio ze sobą przy pomocy szeregowej linii transmisyjnej oraz wszystkie są zasilane z akumulatorów umieszczonych w pojeździe, co sprawia że konieczne jest stałe fizyczne połączenie pojazdu z joystickiem. Jednak dzięki założonej w początkowej fazie projektu, zasadzie modularności systemu sterowania, system pomimo braku udziału komputera PC oraz modułu ARM, działa poprawnie, spełniając wszystkie pozostałe przyjęte założenia.

7. Podsumowanie

Niniejsza praca porusza podstawowe kwestie związane z tematem sterowania pojazdu. Zaopatrzenie podwozia w napęd oraz w mikroprocesorowy system sterujący nie było łatwym zadaniem. Wymagało to praktycznego zastosowania wiedzy z dziedzin, mechaniki, elektroniki, mikro-informatyki, informatyki oraz transmisji danych. Konieczna także była umiejętność radzenia sobie z ciągłym występowaniem nieoczekiwanych problemów i awarii. Praca przedstawia wszystkie zagadnienia od strony praktycznej, przedstawiając wiedzę techniczną na poziomie koniecznym do zrozumienia specyfiki tematu. Taki styl pracy wynika z faktu iż szczegółowe opracowanie pewnych zagadnień mogłoby stać się tematem oddzielnych prac licencjackich lub magisterskich, z powodu kompleksowości zastosowanych rozwiązań.

Począwszy od pierwszej fazy projektu wszystkie elementy pracy starano się opracować i zbudować własnoręcznie. Do zrealizowania większości czynności wykonanych przy budowie i uruchamianiu pojazdu wykorzystywany był komputer, który stał się nieodzownym narzędziem pracy. W pierwszej kolejności wykonywane były komputerowo projektowane schematy ideowe oraz elektryczne kolejnych elementów. Następnie tworzone były projekty płytek drukowanych poszczególnych podzespołów. Zaowocowało to dużą przejrzystością projektowanych schematów, a także dużą dokładnością wykonanych płytek. W końcowej fazie komputer był używany do stworzenia oprogramowania i zapisania go w pamięci układów.

Praca przy projekcie wymagała budowy wielu wersji układów, i napisania kilku wersji oprogramowania. Wymóg ten był podyktowany niemożnością budowy finalnej wersji wszystkich układów elektronicznych, bez wcześniejszego zapoznania się z praktycznym działaniem poszczególnych elementów układów. Proces budowy, ewolucji oraz testowania układów został dokładnie pokazany w poszczególnych rozdziałach pracy.

8. Bibliografia

[1] Nowa encyklopedia powszechna PWN, pod red. Barbary Petrozolin-Skowrońskiej, Warszawa, Wydawnictwo naukowe PWN, 1996, ISBN: 83-01-11097-X

[2] Wikipedia Wolna Encyklopedia, http://wikipedia.pl

[3] N. Pinckney, MicroToys Guide, 2005

[4] Prof. dr hab. inż. Zygmunt Wróbel, Elektronika analogowa, Konspekt wykładu, Sosnowiec, 1999

[5] Prof. dr hab. inż. Jan Piecha, Transmisja danych i sieci komputerowe, Wykład 1 – Elementy transmisji danych, http://zsk.tech.us.edu.pl/dydaktyka/tele/TDiSK1.pdf

[6] Paul Horowitz, Winfield Hill, Sztuka elektroniki w przekładzie Bogusława Kalinowskiego i Grażyny Kalinowskiej, Wyd.2, Warszawa, Wydawnictwo Komunikacji i Łączności

[7] Jarosław Doliński, Mikrokontrolery AVR w praktyce, Wydawnictwo BTC, Wyd.3, Warszawa, 2003, ISBN: 83-910067-6-X

[8] Robert Krysztof, AVR-GCC w praktyce..., http://avr.elektroda.eu/?q=node/8

[9] Guido Socher, Programming the AVR Microcontroller with GCC, 2004, http://www.linuxfocus.org/English/March2002/article231.shtml

[10] Atmel Corporation, AVR 8-Bit RISC, http://atmel.com/dyn/products/param_table.asp?family_id=607&OrderBy=part_no&Direction=ASC

[11] Atmel Corporation, AVR 8-Bit RISC - Application Notes, http://atmel.com/dyn/products/app_notes.asp?family_id=607

[12] Atmel Corporation, Noty katalogowe układów AVR 8-Bit RISC, http://atmel.com/dyn/products/datasheets.asp?family_id=607#760

[13] Nota katalogowa układu LM358, http://focus.ti.com/docs/prod/folders/print/lm358a.html

[14] Strona internetowa: www.olimex.com/dev/

[15] Forum internetowe: http://forum.sparkfun.com/

[16] EP930x quickstart guide, instrukcja dostarczona przez producenta układu na płycie cd-rom