

# **Poradnik „Do It Yourself” do robota sterowanego za pomocą maty.**

1. Wstęp.
2. Przygotowanie.
3. Robot.
  - 3.1. Potrzebne części.
  - 3.2. Budowa robota od zera z wcześniej przygotowanych części. Kod programu do robota.
    - 3.2.1. Gotowy do użytku.
4. Mata Sterująca.
  - 4.1. Potrzebne części.
  - 4.2. Budowa maty od zera z wcześniej przygotowanych części.
  - 4.3. Kod programu do maty.
    - 4.3.1. Gotowy do użytku.
5. Połączenie robota z matą za pomocą programu.
  - 5.1. Gotowy do użytku.
6. Podsumowanie.
  - 6.1. Sugestie dla użytkownika.

## 1. Wstęp.

Cześć, jesteśmy grupą studentów, którzy na swój projekt inżynierski postanowili stworzyć niedużego robota sterowanego za pomocą interaktywnej maty.

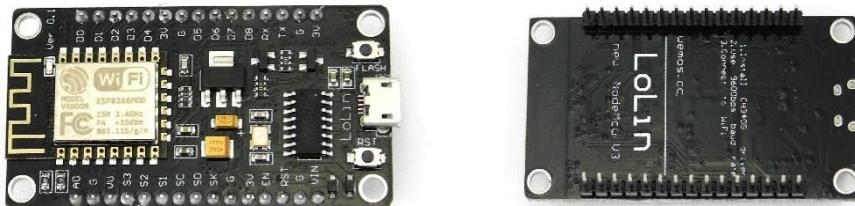
Robot sterowany jest za pomocą sekwencji, które wprowadzamy na w/w macie. Możemy dawać mu proste komendy pozwalające na poruszanie się. Alternatywnym trybem jest jazda swobodna pozwalająca na sterowanie robotem w czasie rzeczywistym. Mata jest urządzeniem posiadającym układ przycisków 4x4 i ekran pokazujący aktualnie wpisywaną sekwencję.

Posiadając podstawową wiedzę na temat programowania i elektroniki, będziesz mógł skonstruować taki zestaw sam, w zaciszu swojego domu. Zalecamy z zapoznaniem się dokładnie z poradnikiem przed próbą konstrukcji. Powodzenia!

## 2. Przygotowanie.

Do utworzenia naszego projektu, będą potrzebne:

### NodeMCU(Arduino)



**ESP8266** jest popularnym mikrokontrolerem z wbudowanym WiFi. Dzięki atrakcyjnej cenie i dużym możliwościami stał się bardzo popularny, zwłaszcza w zakresie automatyki domowej.

Moduł **NodeMCU** to **Arduino** z wlutowanym **ESP8266-12**. Arduino pozwala nam sterować pozostałymi częściami elektronicznymi w projekcie a ESP zapewnia nam bezprzewodową łączność. W naszym projekcie mikrokontroler będzie wykorzystany do sterowania robotem oraz do tworzenia sieci WiFi, do której będzie łączyć się nasza mata.

## Arduino IDE

Natomiast do napisania odpowiedniego oprogramowania przyda nam się **Arduino IDE**, które pozwala zaprogramować nam NodeMCU językiem C. Do poprawnego działania IDE z NodeMCU potrzebne są dodatkowe biblioteki co opiszemy w dalszej części poradnika.

Program ten jest w pełni darmowy i można go uruchomić na Windowsie, Linuxie lub Mac-Osie. Dostępny jest na przykład tutaj:

<https://www.arduino.cc/en/Main/Software>

### Download the Arduino IDE

The screenshot shows the official Arduino Software (IDE) download page. On the left, there's a large teal circular icon with the Arduino logo (an infinity symbol containing a minus and plus sign). To its right, the text "ARDUINO 1.8.10" is displayed in bold. Below it is a short description: "The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. Refer to the [Getting Started](#) page for Installation instructions." On the right side, there are download links for different operating systems: "Windows Installer, for Windows XP and up" and "Windows ZIP file for non admin install"; "Windows app" (Requires Win 8.1 or 10, with a "Get" button); "Mac OS X 10.8 Mountain Lion or newer"; and "Linux 32 bits", "Linux 64 bits", "Linux ARM 32 bits", and "Linux ARM 64 bits". At the bottom right, there are links for "Release Notes", "Source Code", and "Checksums (sha512)".

Wcisnąć interesujący nas link z installerem z prawej strony i przechodzimy dalej. Ja wykorzystam tutaj instalator dla systemu Windows.

### Contribute to the Arduino Software

Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). [Learn more on how your contribution will be used.](#)

The screenshot shows the Arduino Software contribution page. It features a cartoon illustration of three electronic components (a red square, a grey rectangle, and a blue circle) connected by wires. Below the illustration, there is a block of text: "SINCE MARCH 2015, THE ARDUINO IDE HAS BEEN DOWNLOADED 37,530,684 TIMES. (IMPRESSIVE!) NO LONGER JUST FOR ARDUINO AND GENUINO BOARDS, HUNDREDS OF COMPANIES AROUND THE WORLD ARE USING THE IDE TO PROGRAM THEIR DEVICES, INCLUDING COMPATIBLES, CLONES, AND EVEN COUNTERFEITS. HELP ACCELERATE ITS DEVELOPMENT WITH A SMALL CONTRIBUTION! REMEMBER: OPEN SOURCE IS LOVE!" At the bottom, there are several buttons for different contribution amounts: "\$3", "\$5", "\$10", "\$25", "\$50", and "OTHER".

JUST DOWNLOAD

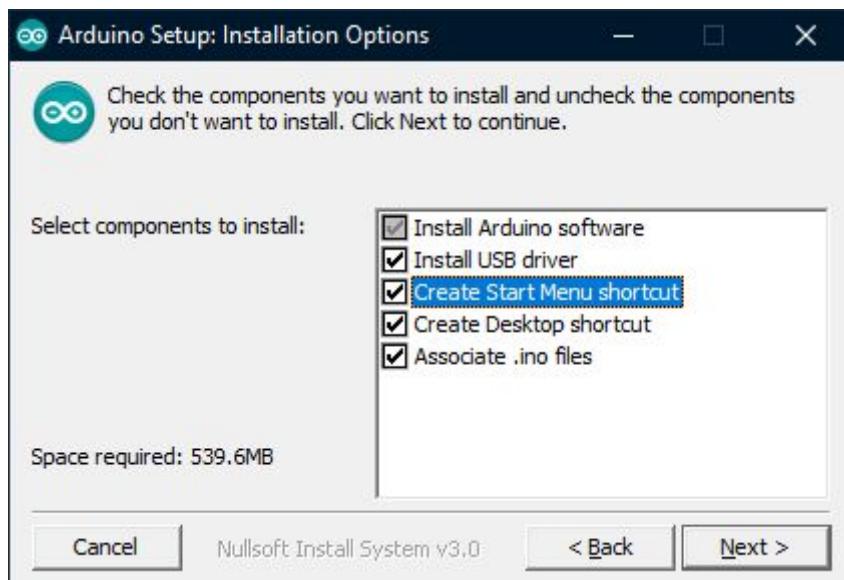
CONTRIBUTE & DOWNLOAD

Jak wspomniałem, program jest w pełni darmowy dlatego strona proponuje nam dobrowolny podarunek pieniężny, więc jeżeli czujesz że chcesz wspomóc twórców - śmiało zaznacz interesującą Cię kwotę i przejdź dalej przyciskiem "**CONTRIBUTE & DOWNLOAD**". Ja natomiast przejdę do pobierania wciskając przycisk "**JUST DOWNLOAD**".

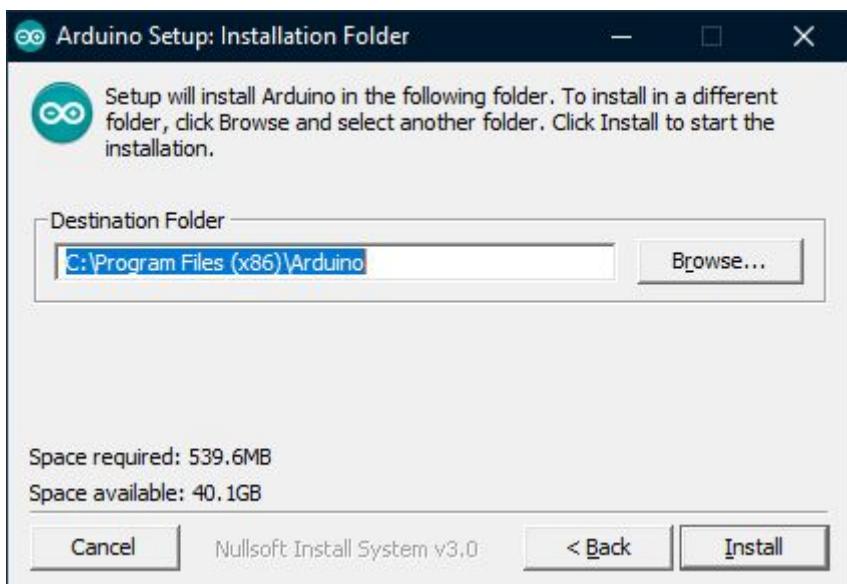
Czekamy aż instalka skończy się pobierać i przechodzimy do instalacji programu. Oczywiście czytamy całą umowę licencyjną i jeśli nie mamy zastrzeżeń wciskamy "I Agree".



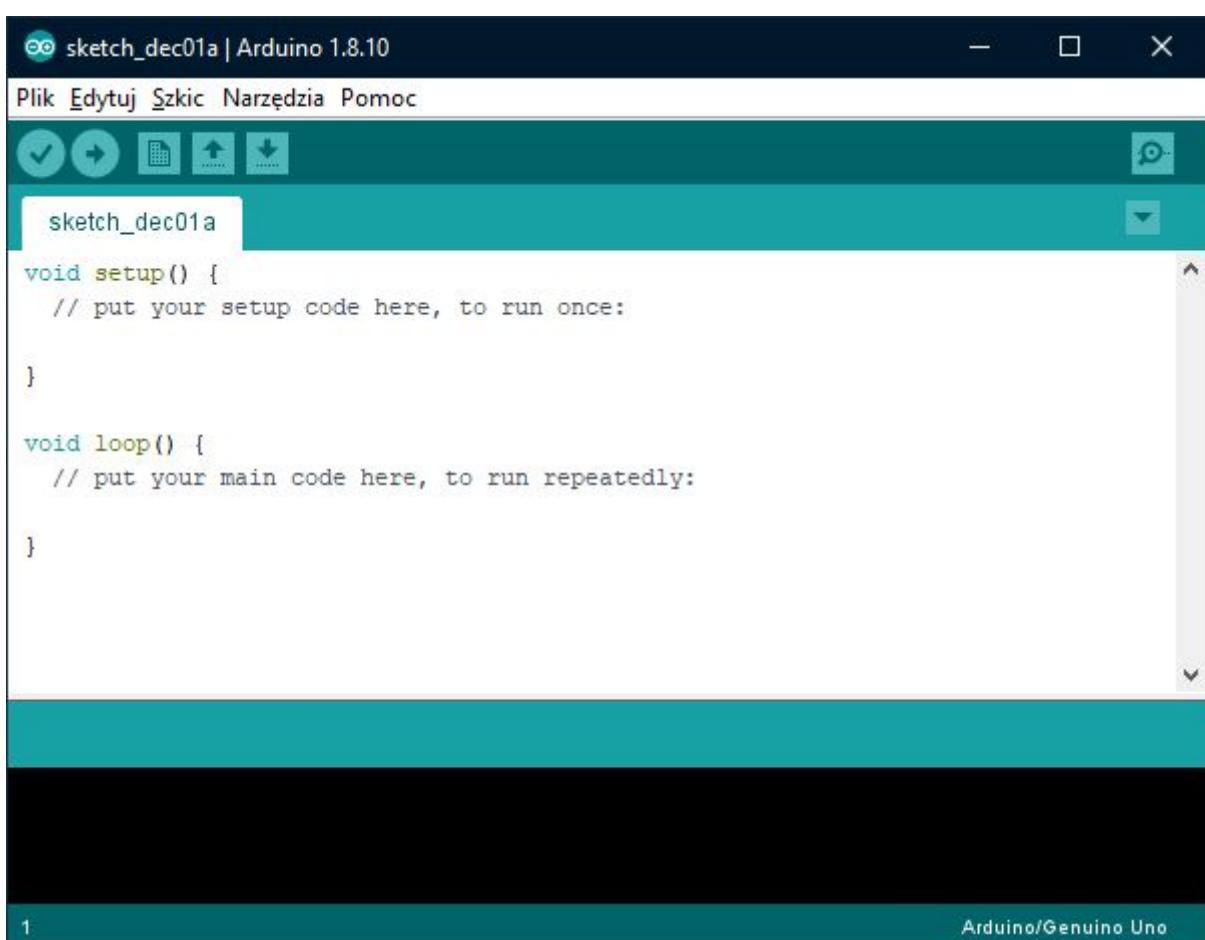
Wybieramy zawartość, którą chcemy zainstalować i przechodzimy dalej.



Następnie wybieramy folder, do którego chcemy zainstalować nasze IDE.

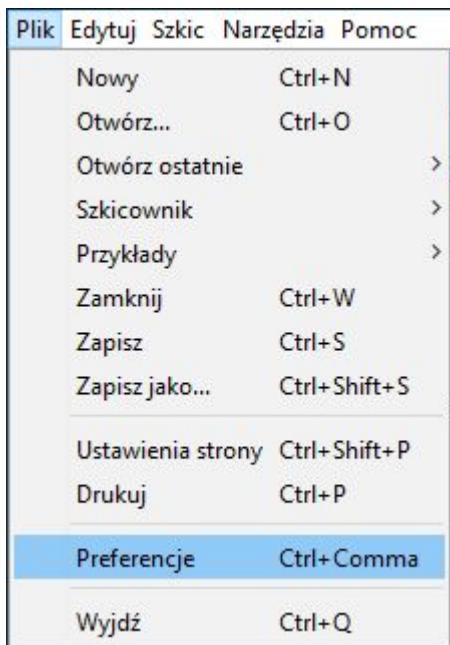


Po zakończonej instalacji zamykamy installer i nasze IDE jest już prawie gotowe do użytku. Musimy je jeszcze odpowiednio skonfigurować.

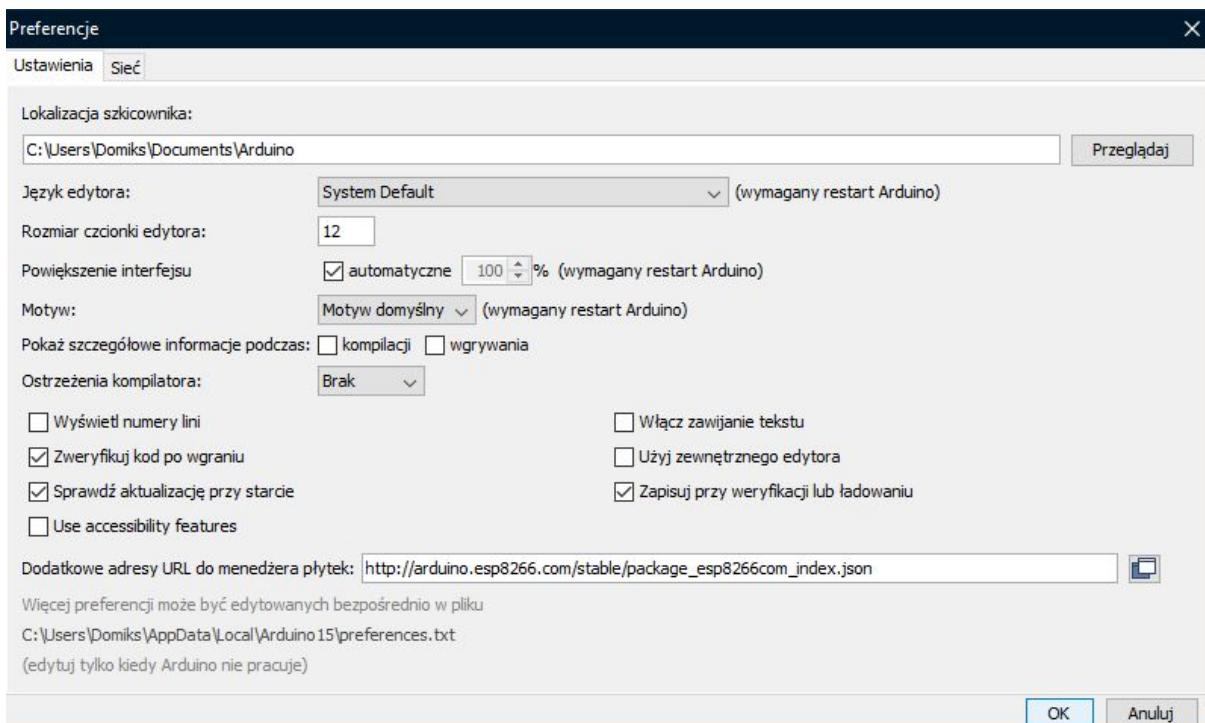


## Konfiguracja IDE

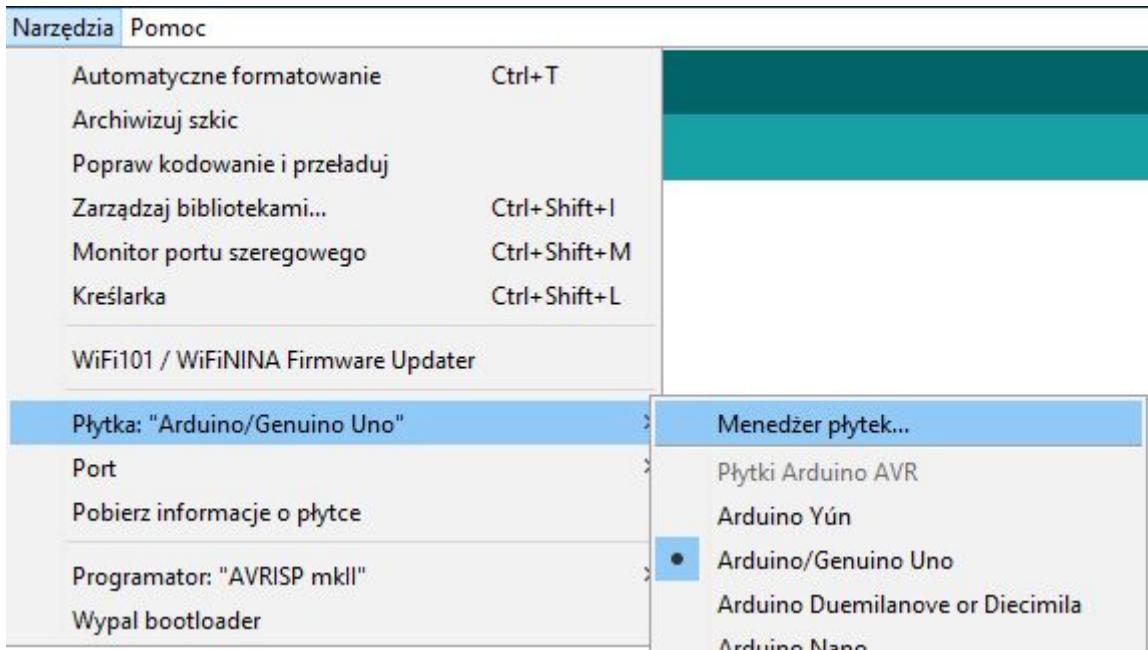
Zaczynamy od włączenia IDE, jeśli zostało przez nas wyłączone. Następnie otwieramy menu preferencji.



Kopiujemy podaną linie i wklejamy do pola "Dodatkowe adresy URL do menedżera płytek" [http://arduino.esp8266.com/stable/package\\_esp8266com\\_index.json](http://arduino.esp8266.com/stable/package_esp8266com_index.json) i potwierdzamy przyciskiem OK.



Kolejnym krokiem jest przejście do menedżera płytek.



W polu wyszukiwania wpisujemy “esp8266” a następnie instalujemy dany pakiet.



Po zakończonej instalacji pakietu, zamykamy okno menedżera płyt i klikamy jeszcze raz w narzędzia i wybieramy “Generic ESP8266 Module” z listy dostępnych płyt.

Narzędzia | Pomoc

Automatyczne formatowanie	Ctrl+T
Archiwizuj szkic	
Popraw kodowanie i przekładaj	
Zarządzaj bibliotekami...	Ctrl+Shift+I
Monitor portu szeregowego	Ctrl+Shift+M
Kreślarka	Ctrl+Shift+L
WiFi101 / WiFiINA Firmware Updater	
Płytki: "Generic ESP8266 Module"	
Builtin Led: "2"	
Upload Speed: "115200"	
CPU Frequency: "80 MHz"	
Crystal Frequency: "26 MHz"	
Flash Size: "1MB (FS:64KB OTA:~470KB)"	
Flash Mode: "DOUT (compatible)"	
Flash Frequency: "40MHz"	
Reset Method: "dtr (aka nodemcu)"	
Debug port: "Disabled"	

Menedżer płyt... ▲

- Arduino BT
- LilyPad Arduino USB
- LilyPad Arduino
- Arduino Pro or Pro Mini
- Arduino NG or older
- Arduino Robot Control
- Arduino Robot Motor
- Arduino Gemma
- Adafruit Circuit Playground
- Arduino Yún Mini
- Arduino Industrial 101
- Linino One
- Arduino Uno WiFi

ESP8266 Boards (2.6.3)

● Generic ESP8266 Module

Kiedy już wybierzemy odpowiedni moduł płytki nasze IDE jest w pełni skonfigurowane i gotowe do użytku.

## Raspberry Pi Zero



Minikomputer Raspberry Pi w wersji miniaturowej. Wyposażony w procesor Broadcom BCM2835 1 GHz i 512 MB pamięci RAM, WiFi, Bluetooth, port miniHDMI, gniazdo microUSB OTG, 40 GPIO, złącza na kartę microSD oraz cztery otwory montażowe. Wymiary płytki to: 65 x 30 x 5 mm. Ta wersja posiada wlutowane złącza męskie GPIO.

Nasza "malinka" będzie odpowiedzialna za sterowanie matą.

Przykładowy link do sklepu:

<https://botland.com.pl/pl/moduly-i-zestawy-raspberry-pi-zero/9749-raspberry-pi-zero-wh-512mb-ram-wifi-bt-41-ze-zlaczami.html>

Niestety, w chwili pisania poradnika, malinka dostępna jest tylko na botlandzie lub allegro.

### Instalacja Node.js

Zaczynamy od wybrania odpowiedniej wersji Node'a, w naszym przypadku jest to wersja [v8.9.1](#) i wybieramy plik "linux-armv6l", ponieważ używamy Raspberry PI Zero. Łączymy się z malinką za pomocą SSH i wpisujemy daną komendę, która pobierze nam odpowiednią wersję Node'a.

```
curl -o node-v8.9.1-linux-armv6l.tar.gz  
https://nodejs.org/dist/v8.9.1/node-v8.9.1-linux-armv6l.tar.gz
```

Kiedy już zakończymy pobieranie, musimy wypakować pliki używając do tego komendy

```
tar -xzf node-v9.7.1-linux-armv6l.tar.gz
```

Utworzy się nowy folder zawierający wszystkie potrzebne pliki do Node.js oraz Node Package Manager (NPM). Następnie wpisujemy komendę, która skopiuje wszystkie wypakowane pliki do odpowiedniego miejsca.

```
sudo cp -r node-v9.7.1-linux-armv6l/* /usr/local/
```

Po wykonaniu tej komendy, wszystko powinno być na swoim miejscu. W tym momencie możesz już zacząć używać Node.js. Dla pewności, można jeszcze wykonać komendę

```
node -v  
npm -v
```

Warto tutaj dodać, że wspomniany wcześniej NPM opiera się mocno o dostęp do Gita. Domyslnie Linuxowy Raspian nie nie ma zainstalowanego dostępu do Gita, dlatego warto jeszcze wykorzystać poniższą komendę.

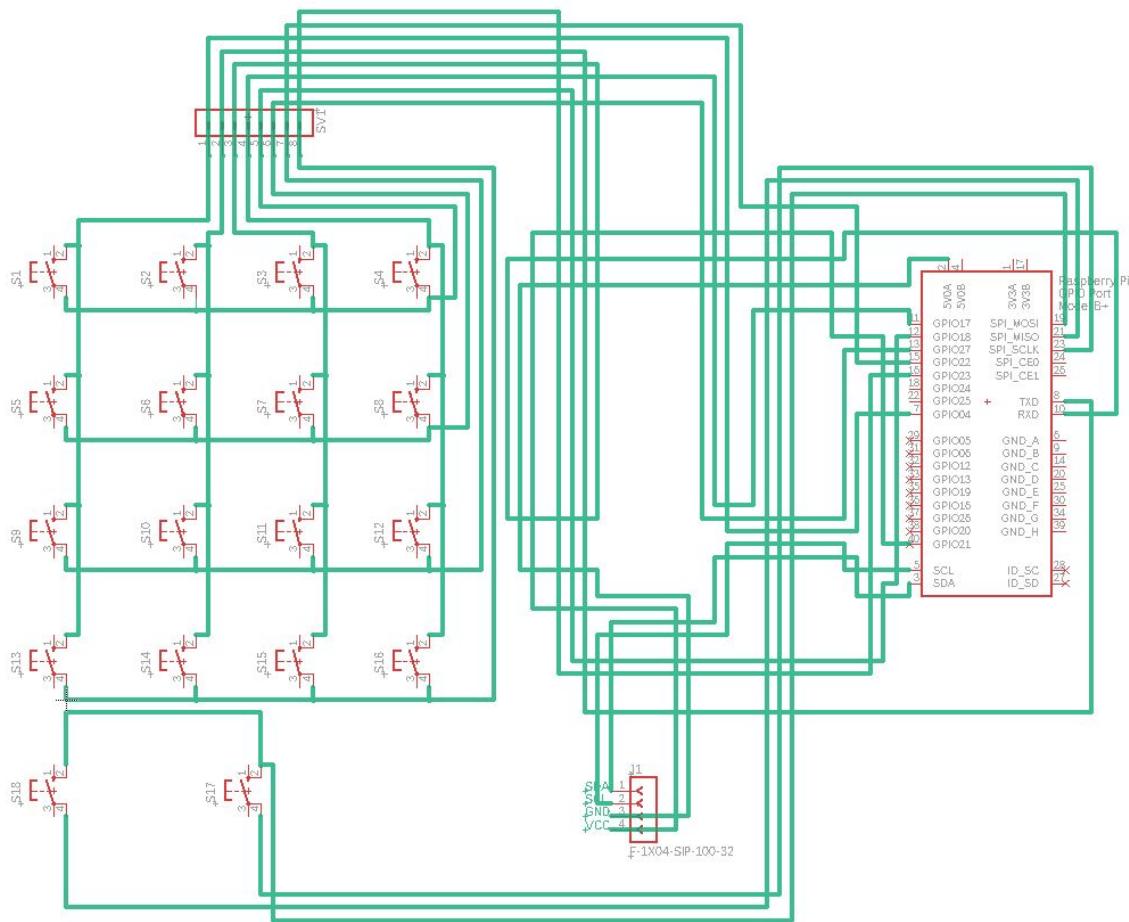
```
sudo apt-get install git
```

W tym momencie wszystko powinno być skonfigurowane i gotowe do użytku.

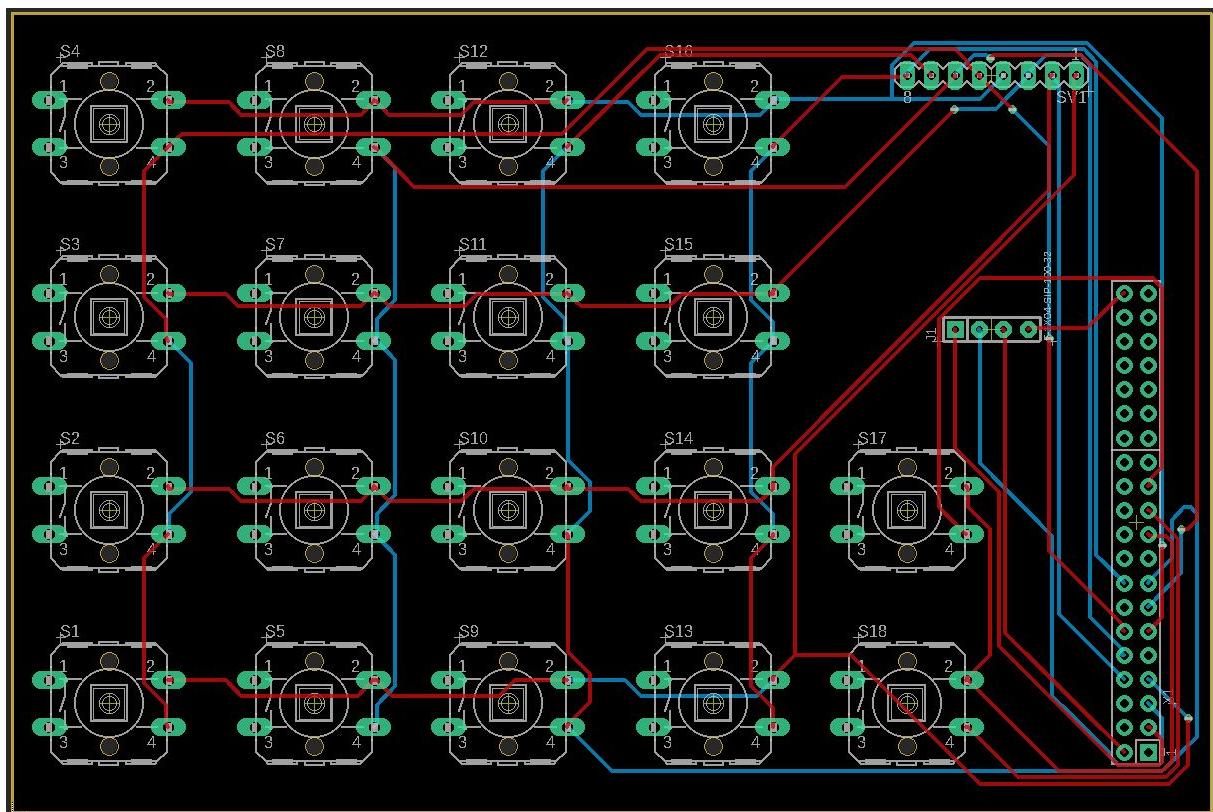
### Części do budowy robota oraz maty

Oczywiście do budowy naszego projektu będą potrzebne różne części, które opiszemy w kolejnych rozdziałach.

### Płytki PCB



Zdjęcie przedstawia schemat połączeń w macie. Przyciski, ekran oraz k넥ktor do Raspberry Pi



*Zdjęcie przedstawia projekt płytki drukowanej dwustronnej. Czerwone linie oznaczają ścieżki na górnej warstwie laminatu, niebieskie na dolnej warstwie.*

Do naszego robota oraz maty będzie potrzeba po jednej płytce pcb. W tym miejscu mamy dwa sposoby na ich przygotowanie. Możesz je przygotować samemu, tak jak zrobiliśmy to my lub możesz takowe płytki zamówić za dodatkową opłatą. Jeśli zdecydujesz się zamówić płytki, będzie Ci potrzebny jedynie plik ze schematem, który musisz wysłać do odpowiedniej firmy.

[Pliki do schematu maty](#)

[Pliki do schematu robota](#)

Przykładowe linki do firm:

1. <https://jlpcb.com/> - czas wykonania, około 3 tygodnie
2. <https://www.pcbway.com/> - czas wykonania zależy od obliczenia, mniej więcej 1,5 tygodnia
3. <https://www.fabrykapcb.pl/index.html> - czas wykonania, około tygodnia

Przejdźmy teraz do przygotowania płytek w warunkach domowych.

## Przygotowanie

Potrzebne Ci będzie kilka rzeczy,

- **płyta grzejna lub żelazko**, najlepiej takie bez dziurek z płaską powierzchnią, inaczej płytka może się wygiąć,
  - **projekt PCB**, który dostaniesz od nas
  - najistotniejsza będzie jednak **drukarka laserowa**, jeśli jednak nie masz, popytaj znajomych, w punktach ksero najczęściej odmawiają - jest ryzyko przyklejenia się papieru do głowicy,
  - **cienki papier kredowy**  
<https://www.ebiuromax.pl/papier-kredowy-a4-135g-50-błyszczący.html>  
<https://biurwa.pl/papier-kredowy-kreska-a4-130g>
  - mała **wiertarka modelarska**, najlepiej z **tytanowymi** wiertłami o szerokości **1mm**,  
<https://botland.com.pl/pl/wiertarki-i-wkretarki/665-miniwiertarka-velleman-vthd01-z-akcesoriami-5410329377823.html>  
<https://abc-rc.pl/product-pol-7932-Mini-Wiertarka-modelarska-Szlifierka-12V-Velleman-VTHD01.html>
  - **kwas b327** (nadsiarczan sodowy),  
<https://botland.com.pl/pl/wytrawiacze/1057-wytrawiacz-b327-100g-5901764329183.html>  
<https://abc-rc.pl/product-pol-10124-Wytrawiacz-do-płytek-drukowanych-B327-Nadsiarczan-sodowy.html>
  - **aceton**,  
<https://www.ichemia.pl/37.aceton-99-9-hybrydy-250-ml.html>
- aceton możemy kupić spokojnie na allegro - nie ma zbyt dużego wyboru, potrzebujemy czysty aceton, w niewielkiej ilości, wystarczy nawet 100ml.

Zamiast acetonu możemy użyć **IPA - alkohol izopropylowy**.

<https://botland.com.pl/pl/smarty-i-oczyszczacze/974-kontakt-ipa-plus-oliwiarka-100ml-5901764329930.html>

<https://abc-rc.pl/product-pol-9033-Izopropanol-AG-Kontakt-IPA-plus-100ml.html>

Ostatecznie możemy użyć zmywacza do paznokci zawierającego aceton. Dzisiaj często zmywacze są z odżywkami i zawierają jedynie śladowe ilości acetonu, ale jeśli znajdziemy/mamy dostęp do takiego bez odżywek, będzie wystarczająco dobry do wyczyszczenia płytka.

- **miedziane tulejki 0.9mm**,

<https://www.gotronik.pl/tulejki-miedziane-0-90mm-do-wykonywania-przelotek-p-4945.html>

tulejki możemy również kupić na allegro - trzeba jedynie pamiętać aby były **miedziane** o średnicy **0.9mm**

- przyda się również **gąbka druciana** lub **papier Ścierny** o gradacji (ziarnistości) większej niż 600

<https://www.castorama.pl/papier-wodoodporny-condor-230-x-280-mm-gradacja-800-id-85959.html>

<https://www.leroymerlin.pl/narzedzia-reczne/materialy-scierne-szczotki-druciane/materialy-scierne-ciete-i-w-arkuszach/papier-scierny-wodny-p800-230-x-280-mm-dexter.p453983,l810.html>

- **marker permanentny**,

<https://botland.com.pl/pl/markery/952-marker-permanentny-czarny-pentel-n850-4902506071361.html>

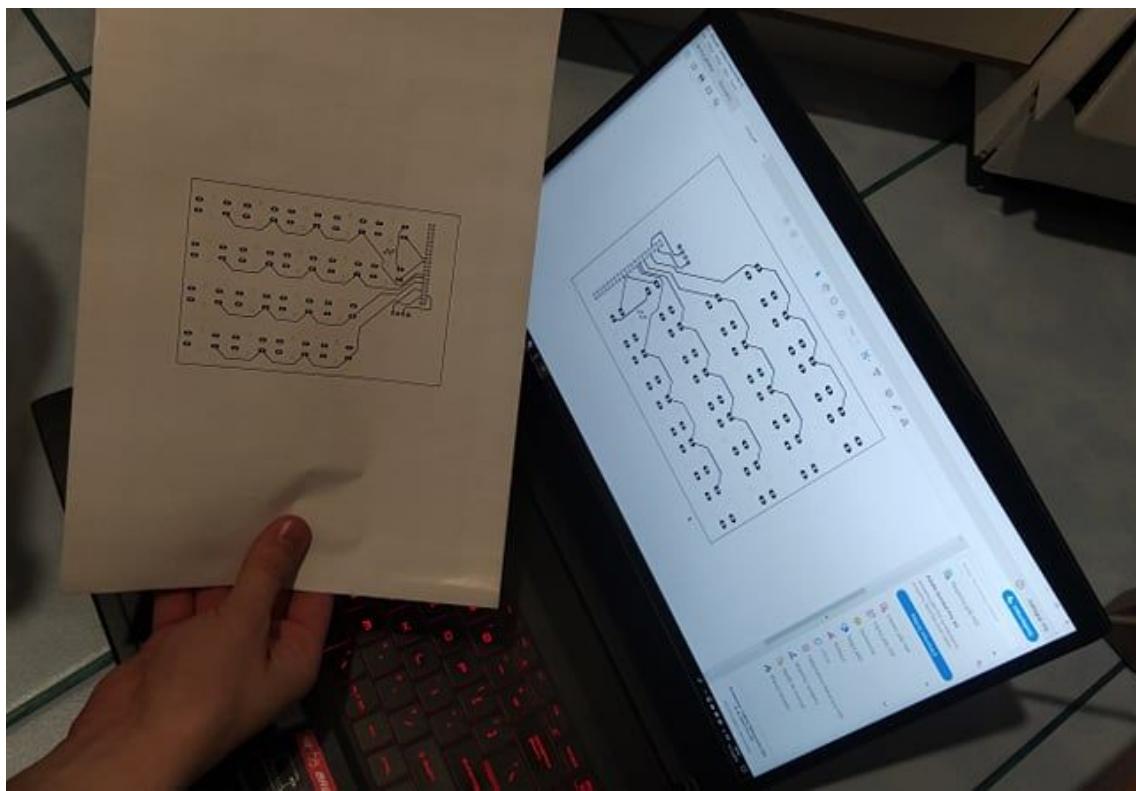
<https://abc-rc.pl/product-pol-9594-Marker-do-rysowania-sciezek-0-3mm-czarny.html>

- **gumowe rękawiczki** - kwas, który będziemy używać nie jest groźny przy kontakcie ze skórą. Nie zmienia to faktu, że lepiej zachować ostrożność.

Zanim przejdziesz już do tworzenia płytki, pamiętaj aby zachować wszelką ostrożność! Powinieneś znajdować się w dobrze wentylowanym pomieszczeniu.

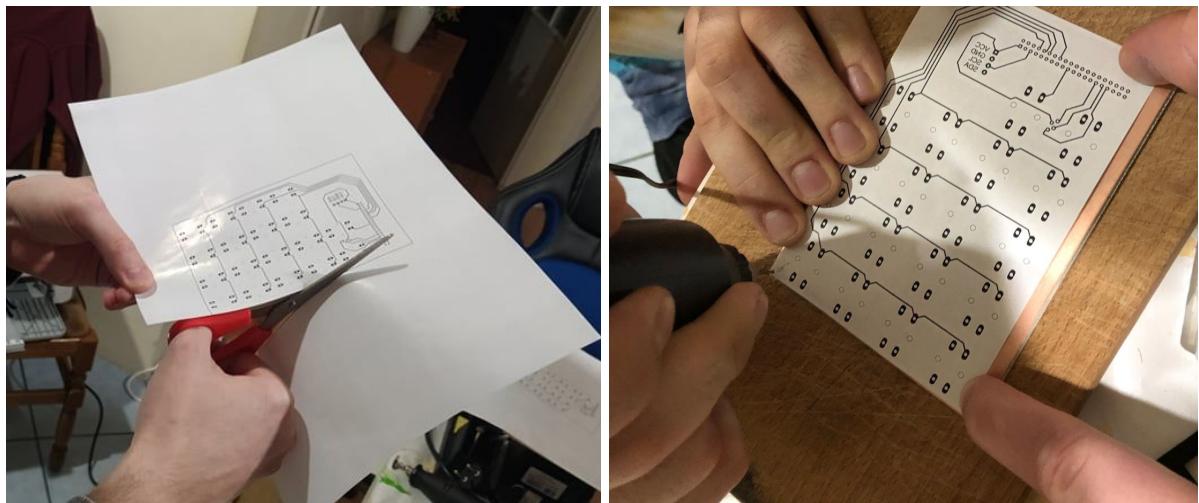
## Drukowanie

W tym kroku jest nam potrzebna drukarka laserowa. Jak wspomniałem wyżej, jeśli masz - świetnie, jeśli jednak nie, popytaj znajomych. Musimy wydrukować schemat płytki, najlepiej żebyś to zrobił przy użyciu papieru kredowego - jest tani i łatwo dostępny, także nie powinieneś mieć problemu z dostaniem go.



## Wymiarowanie

Teraz przykładamy schemat do płytki i wiercimy dziurki w rogach schematu na płytce, tak aby góra i dół pokrywały się ze sobą, ponieważ płytką będzie dwustronna. Następnie docinamy płytkę do wielkości schematu i przechodzimy do kolejnego kroku.





### Czyszczenie powierzchni

Powierzchnię płytki należy oczyścić przed wykonaniem kolejnych czynności. Odbywa się to w dwóch krokach. Najpierw szlifujemy lekko powierzchnię z obu stron płytki używając papieru ściernego lub druciaka. Po zakończonym procesie szlifowania przecieramy powierzchnię alkoholem bądź acetonem - my użyliśmy do tego starej szczoteczki do zębów maczanej w czystym acetonie.



### Nakładanie Ścieżek

Zaczynamy od rozgrzania żelazka, musi być ustawione na maksymalną temperaturę. Przyda się tutaj jakaś płaska powierzchnia jak na przykład kafelki w łazience czy kuchni albo drewniana deska do krojenia. Układamy kawałek papieru ręcznikowego lub toaletowego. Na papierze układamy płytke, tak aby nie dotykała kafelek. Następnie na płytke kładziemy papier kredowy z wydrukowanym schematem i ustawiamy zgodnie z wyrobionymi dziurkami. Toner powinien być ustawiony do dołu.



Dokładamy kolejny kawałek papieru ręcznikowego, uważając żeby schemat na płytce nie przesunął się. Kiedy już wszystko jest ładnie ustawione, dociskamy całość rozgrzonym żelazkiem i co kilka sekund wykonujemy ruch prasowania. Cały proces powinien trwać około 5 minut.



Po zakończonym procesie nakładania, ostrożnie wyciągamy płytke (powinieneś tu zachować ostrożność, całość może być dosyć gorąca). Następnie płytke z przyklejonym już schematem ostrożnie odkładamy w misce i chłodzimy pod bieżącą wodą, przy okazji warto namoczyć przyklejony schemat.

Kiedy schemat będzie już dobrze namoczony, powoli odrywamy go od płytki. Na naszej płytce powinien

być obecnie widoczny toner ze ścieżkami, jeżeli dostrzegasz jakieś przerwane ścieżki, możesz je ostrożnie poprawić markerem permanentnym.

**Cały proces powtarzamy dla drugiej strony laminatu. Wywiercone wcześniej dziury pozwolą ułożyć wydruk tak, aby pokrywał się z drugą stroną.**

## Wiercenie

Musimy teraz wywiercić małe otwory na piny. Używamy do tego naszej wiertarki modelarskiej z tytanowym wiertłem o szerokości 1mm. Następnie w powstałe otwory wbijamy miedziane tulejki.

**~~przyda się jakieś zdjęcie~~**

## Wytrawianie

W tym miejscu powinieneś założyć swoje gumowe rękawiczki. Sam roztwór nie powinien być niebezpieczny, jednak ostrożności nigdy za wiele!

Przejdziemy teraz do utworzenia roztworu, który wytrawi miedź z płytki. Przygotuj 100 gramów kwasu B327 oraz pół litra wody (takie są proporcje roztworu), najlepiej o temperaturze 50 stopni, jeżeli nie jesteś w stanie kontrolować temperatury wody możesz ostatecznie użyć wrzątku. Zalewasz kwas ciepłą wodą i mieszasz wszystko do momentu, w którym kryształki kwasu się rozpuszczą.

Do utworzonego roztworu wrzucamy naszą płytke. Całość najlepiej umieścić w płaskim pudełku. Kiedy płytka znajduje się w środku, potrzasaj lekko pudełkiem, tak aby mieszanka była cały czas w ruchu, ponieważ to przyspiesza proces trawienia. Całość powinna trwać od 5 do 10 minut. Po upływie około 3 minut, obróć płytke, aby zobaczyć jak proces

trawienia przebiega z drugiej strony.



Kiedy miedź zostanie usunięta z miejsc, w których nie ma toneru, proces trawienia można uznać za zakończony. Wyciągamy płytę z mieszanki. Następnie płytę wycieramy i myjemy pod bieżącą wodą.

Warto tutaj dodać, że jeśli planujesz zrobić więcej płyt, to roztwór jest wielokrotnego użytku, jednak z każdym użyciem będzie zmieniał kolor na coraz bardziej niebieski i proces trawienia będzie odrobinę wolniejszy.

Sprawdzamy teraz czy wszystkie ścieżki wyglądają poprawnie, jeśli tak to zmywamy toner acetonom. Miejsca, w których toner przykleił się bardzo mocno, można lekko przetrzeć papierem ściernym lub drucianą gąbką.

### Pistolet do kleju



Zalecane właściwości:

- Moc - 10 W,
- Średnica kleju - 8 mm,

Przykładowy link:

<https://abc-rc.pl/product-pol-11945-Pistolet-do-kleju-na-goraco-110-240V-40W-Glue-Gun.html> +

<https://abc-rc.pl/product-pol-11946-Klej-do-pistoletu-na-goraco-16x1-1cm-klej-do-klejenia-na-goraco-przezroczysty.html>

<https://botland.com.pl/pl/kleje-i-klejarki/976-pistolet-do-klejenia-20w-zd-5.html>

<https://botland.com.pl/pl/kleje-i-klejarki/896-klej-do-pistoletu-cienki-74150-mm-przezroczysty.htmlv>

## Lutownica kolbową



Zalecane właściwości:

- Napięcie zasilania - 230 V,
- Moc - 60 W,

Przykładowy link do sklepu:

<https://botland.com.pl/pl/lutownice-kolbowe/1126-lutownica-kolbowa-zd200c-oporowa-60w.html>

<https://abc-rc.pl/product-pol-12145-Lutownica-220V-60W-regulacja-temperatury.html>

## Śrubokręt i Śrubki

- Śrubokręt płaski
- Śrubokręt krzyżakowy

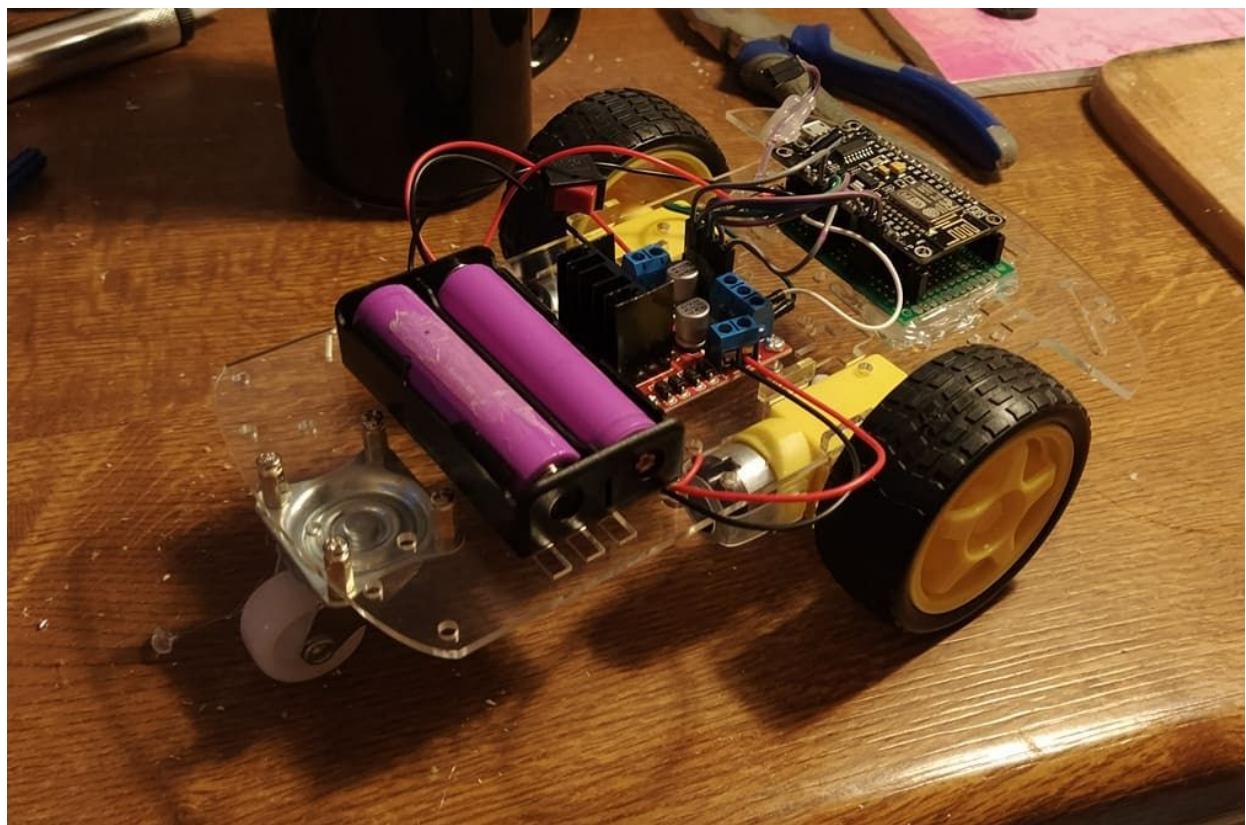
<https://botland.com.pl/pl/zestawy-narzedzi/12640-zestaw-wkretakow-precyzyjnych-16-elementow-5900804106333.htm>

Taki zestaw powinien w zupełności wystarczyć. Naszym zdaniem lepiej kupić taki zestaw niż kupować pojedyncze śrubokręty

- 4 Śrubki M3x12mm
- 4 nakrętki M3

<https://botland.com.pl/pl/srubki-i-nakretki/636-zestaw-srubki-i-nakretki-180szt-5410329304485.html>

## 3. Robot.



## **Lista wymaganych materiałów.**

- 1) Chassis Rectangle 2WD 2-kołowe podwozie robota z napędem – 1 sztuka**



Przezroczysta platforma do budowy robota. W zestawie znajdują się dwa silniki prądu stałego z kołami o średnicy 65 mm. W części przedniej podporę stanowi metalowe koło obrotowe. Elementy podwozia są zrobione z akrylu, posiadają otwory montażowe pozwalające zamontować różnego rodzaju sensory czy kontrolery.

Zestaw zawiera:

- Platformę montażową
- Dwa koła z oponami o średnicy 65 mm
- Dwa silniki prądu stałego z przekładniami
- Jedno metalowe koło obrotowe montowane przy pomocy dwóch śrub
- Metalowe oraz plastikowe podzespoły (śrubki, nakrętki, itp.)
- Włącznik
- Koszyk na baterie (4 x AA) ze przewodami ze zdjętą izolacją

Warto tutaj dodać, że jeśli zdecydujesz się na zakup danego podwozia, to nie musisz już martwić się o silniki oraz koła, ponieważ są zawarte w tym zestawie. Natomiast nadal powinieneś zakupić podany w tej liście koszyk na baterie.

Przykładowy link do sklepu:

<https://botland.com.pl/pl/podwozia-robotow/7283-chassis-rectangle-2wd-2-kolowe-podwozie-robota-z-napedem.html>

<https://abc-rc.pl/product-pol-7656-Podwozie-robota-2WD-RT-5-210mm-2-silniki-podwozie-Arduino.html>

2) **Silnik DC z przekładnią 1:48 – 2 sztuki** (zawarte w zestawie z podwoziem)



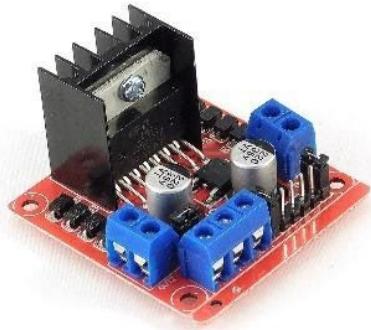
Unikalna jednostka napędowa do Twojego robota. Silnik prądu stałego z przekładnią 1:48 zapewni wspaniałe osiągi dla Twojego robota. Dwustronna oś pozwala zamontować koło z oponą z dowolnej strony.

3) **Koło plastikowe z oponą 65/25 mm – 2 sztuki** (zawarte w zestawie z podwoziem)



Doskonała ochrona przed aquaplaningiem. Krótka droga hamowania. Bardzo dobra przyczepność na zakrętach. Niski poziom hałasu przez cały okres eksploatacji opony.

4) **Moduł dwukanałowego sterownika silników z L298N – 1 sztuka**



Umożliwia on sterowanie jednocześnie dwóch silników DC lub jednego silnika krokowego. Kontroler ten pozwala ustawić stałą (pełną) prędkość obrotu kół, lub kontrolować ją sygnałem PWM.

Przykładowy link do sklepu:

<https://abc-rc.pl/product-pol-6196-Modul-sterownika-L298N-do-silnikow-DC-i-krokwych-Arduino.html>

<https://nettigo.pl/products/modul-dwukanalowego-sterownika-silnikow-z-l298n>

##### 5) Moduł WiFi NodeMCU V3 LoLin ESP-12E(Arduino) – 1 sztuka



Przykładowy link do sklepu:

<https://nettigo.pl/products/modul-wifi-nodemcu-v3-lolin-esp-12e>

<https://botland.com.pl/pl/moduly-wifi/8241-modul-wifi-esp8266-nodemcu-v3.html>

6) **Kondensator 100uF 16V lub więcej** - 1 sztuka



Przykładowy link do sklepu:

<https://botland.com.pl/pl/kondensatory-elektryczne-tht/898-kondensator-elektryczny-100uf35v-6x12mm-105c-tht-10szt.html>

<https://nettigo.pl/products/zestaw-5-kondensatorow-elektrycznych-100-f-35v>

7) **Dwupozycyjny przełącznik MTS-102** – 1 sztuka



Przełącznik dźwigniowy, dwupozycyjny, jednosekcyjny, MTS-1.

Dane techniczne:

Rodzaj napięcia AC/DC Obciążenie do 3A przy 250V Obciążenie do 6A przy 125V

Wysokość: 33mm Średnica otworu montażowego: 6mm

Jeśli dostaniemy koszyk na ognia z włącznikiem, wtedy ta część jest nie potrzebna

Przykładowy link do sklepu:

<https://nettigo.pl/products/dwupozycyjny-przelacznik-mts-102>

<https://botland.com.pl/pl/przelaczniki-z-dzwignia/2478-przelacznik-dzwigniowy-on-off-250v3a.html>

8) **Koszyk na dwa ognia 18650 2S – 1 sztuka**



Łączy w szereg dwa ognia 18650. Daje to od ~7.0 do 8.4 V w zależności od naładowania ogniw. Długi 30-centymetrowy kabel daje się łatwo układać.

Przykładowy link do sklepu:

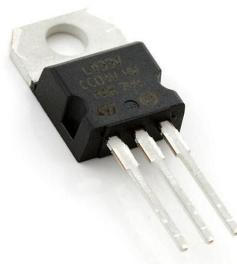
<https://botland.com.pl/pl/koszyki-na-baterie/5240-koszyk-na-2-baterie-typu-18650.html>

<https://abc-rc.pl/product-pol-8528-Koszyk-na-akumulator-2x-18650-3-7V-Li-Ion-koszyczek-na-baterie-ogniwo-z-przewodami.html>

Koszyk z włącznikiem:

<https://abc-rc.pl/product-pol-12801-Koszyk-na-akumulator-2x-18650-z-wylaczniakiem-koszyczek-z-pokrywa-i-przewodami.html>

9) **Stabilizator liniowy 3.3V min. 0.8A – 1 sztuka**



Przykładowy link do sklepu:

<https://nettigo.pl/products/stabilizator-liniowy-3-3v-0-8a>

<https://botland.com.pl/pl/regulatory-napiecia/7685-stabilizator-ido-33v-id1117v33-th-to220.html>

## Budowa Robota

Będąc odpowiednio przygotowanym, możemy przejść do budowy naszego robota.

## Przygotowanie silników

Sprawdzamy czy w silnikach mamy zlutowane kabelki, jeśli są to przejdź do kolejnego kroku.

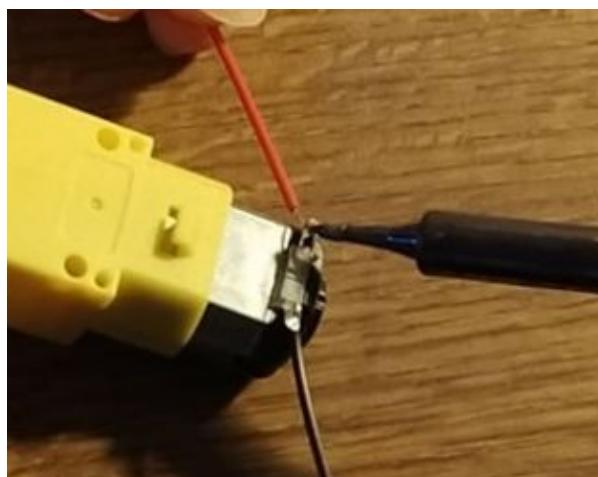
Zdejmujemy najpierw mały kawałek izolacji na kabelkach - około 1 do 2mm, możesz do tego użyć na przykład noża do tapet. Lekko natnij izolację i pociągnij, powinna bez problemu zejść.

Następnie potrzebujemy rozgrzać lutownicę, jeżeli posiadasz zwykłą kolbową - dotknij grotu cyną, jeśli będzie się topić to lutownica jest wystarczająco nagrzana, jeśli jednak posiadasz lutownicę z możliwością regulacji temperatury - ustaw ją w okolicach 400-450 stopni.

Na blaszkę w silniku nakładamy odrobinę cyny.

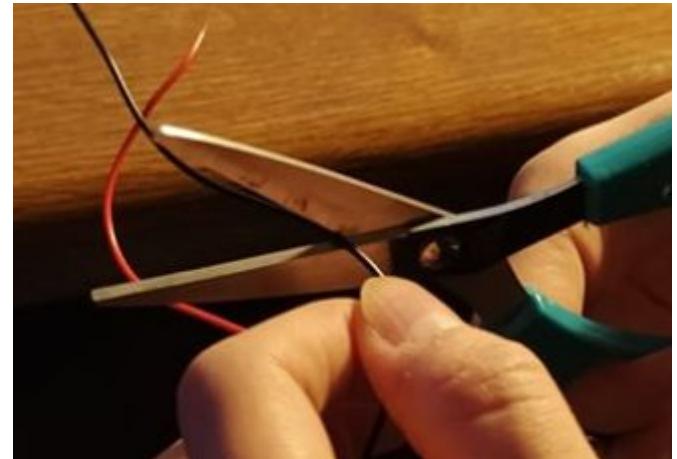
Odsłonięty kabel przykładamy do blaszki silnika - kolor kabelka jest obecnie nieistotny, polaryzację możemy zmienić później w kodzie robota. Przykładamy rozgrzaną lutownicę do odsłoniętego kabelka i nakładamy cynę tak, aby rozgrzała cynę na blaszce i abyśmy mogli przełożyć kabelek przez dziurkę i pokryć go cyną. Oczywiście nie powinieneś ruszać kabelkiem w trakcie lutowania. Po krótkiej chwili lut zastygnie i połączenie będzie gotowe. Powtarzamy tą czynność dla reszty kabelków.

Jeśli korzystasz z koszyka na ogniwa, które podaliśmy w liście części, prawdopodobnie nie będzie musiał lutować kabelków. Jeśli jednak koszyk nie miałby kabli lub miałbyś inny koszyk na baterie, powtórz powyższe czynności.



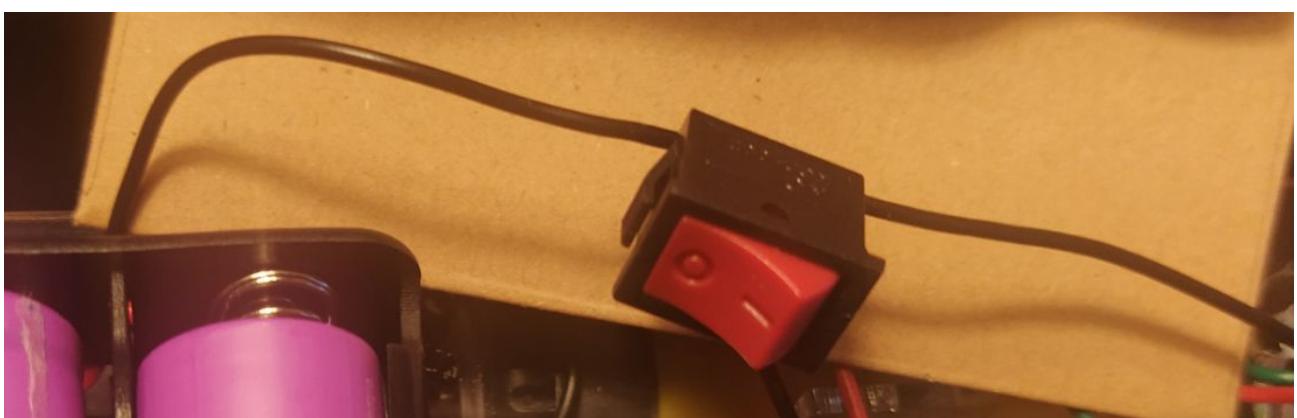
## Lutowanie włącznika

Istotnym jest jeszcze dolutowanie przycisku włączenia/wyłączenia, bez tego nasz robot się nie uruchomi. Łapiemy za koszyk na ognisko i ucinamy czarny kabelek mniej więcej w połowie. Teraz musimy jeszcze zdjąć kawałek izolacji na końcu kabelka od koszyka oraz z obu stron na uciętym kawałku kabelka.



Następnie przechodzimy do lutowania.

Kabelek, który został odcięty lutujemy do blaszki w miejscu, w którym znajduje się "I" na przycisku, a w miejscu "O" powinniśmy dolutować czarny kabel, który wychodzi od koszyka na ogniska. "I" oznacza włączony, a "O" wyłączony.



## Lutowanie płytki

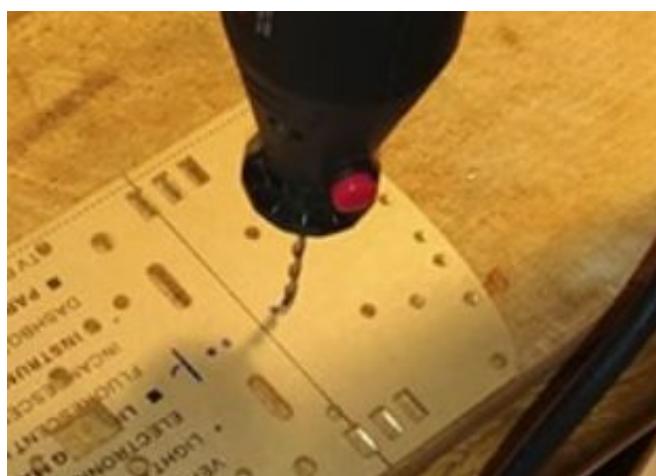
~~DODAĆ OPIS~~

Po zakończonym lutowaniu dobrze jest wgrać kod do robota, tak dla wygody. Opis jak to zrobić znajdziesz w kolejnym dziale nazwanym "Kod do robota".

## Przygotowanie podwozia

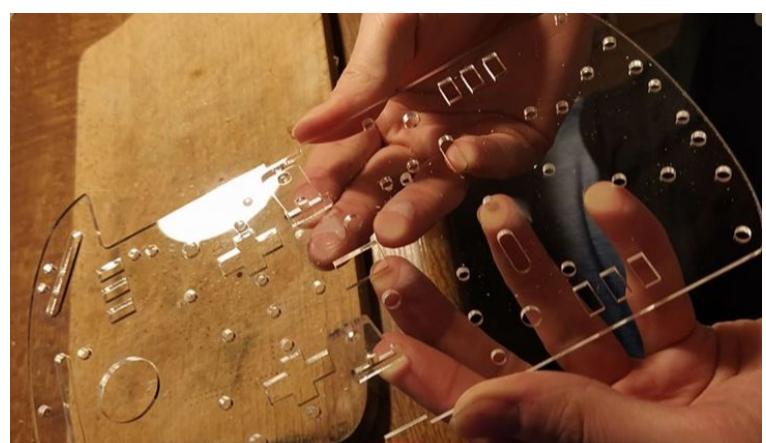


Musimy odpowiednio wywiercić otwory w podwoziu, tak aby móc zamontować części w odpowiednich miejscach. Przykładamy koszyki od akumulatorów i zaznaczamy markerem miejsca, w których następnie wiercimy otwory.



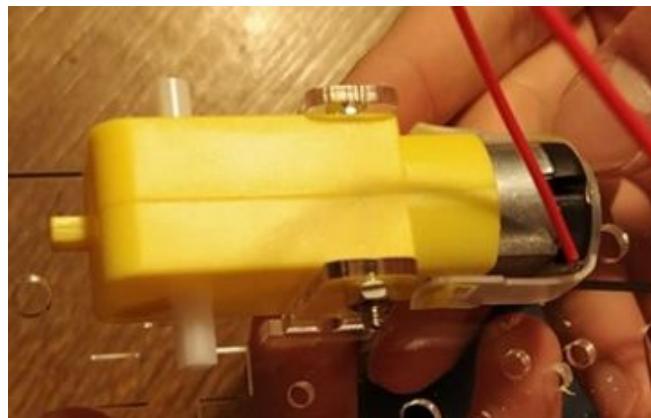
Do wiercenia używamy wiertła 3mm. Czynności powtarzamy dla kontrolera DC.

Na koniec odklejamy jeszcze folię zabezpieczającą podwozie.

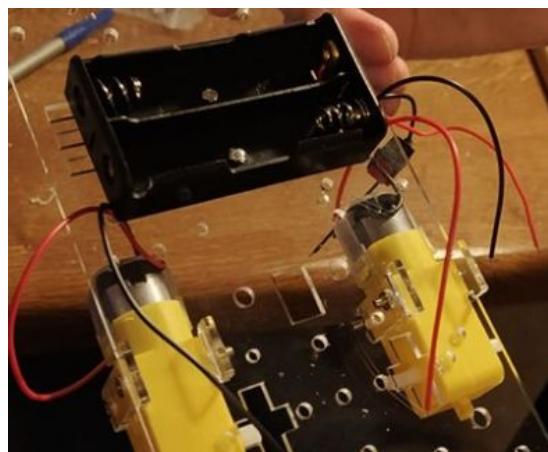
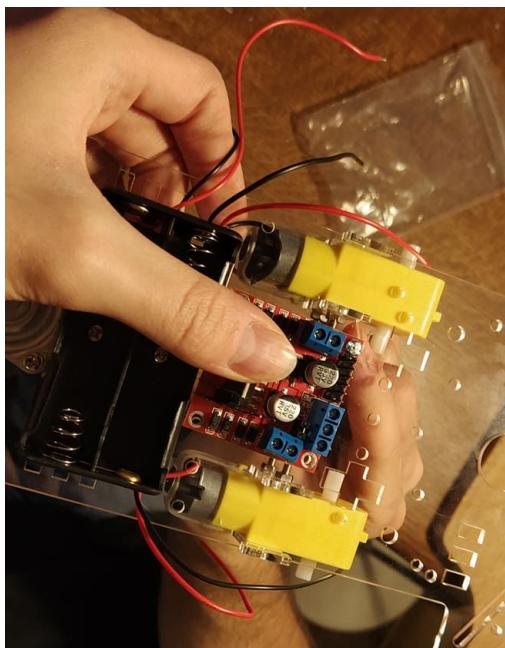


## Montaż części

Łapiemy za śrubokręt krzyżakowy oraz śrubki i przykręcamy do podwozia silniki w uchwytnach montażowych.



Następnie montujemy części, dla których wywierciliśmy otwory w punkcie powyżej, czyli kontroler DC oraz koszyk na ognia.



Przykręcamy również koło obrotowe, dla którego zrobiono miejsca na przodzie podwozia.



Po zamontowaniu wszystkich części, polecamy użyć pistoletu na klej i zakleić koło obrotowe. Dzięki temu, robot będzie bardziej stabilny podczas jazdy. Uważaj tylko żeby przyklejone koło było ustawione prosto, inaczej robot będzie niepotrzebnie skręcał w trakcie jazdy.



### **Połączenie części**

Zaczynamy od połączenia kabli silników do wejść na kontrolerze DC - są to skręcone, skrajne podwójne wejścia. Następnie lutujemy kable odpowiedzialne za zasilanie do płytka od Node'a. Łączymy jeszcze zasilanie Node'a z kontrolerem DC - plus do wejścia 12V a minus do

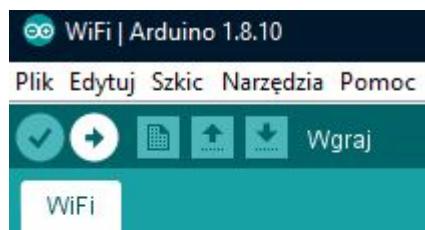
wejścia GND. Na koniec łączymy piny od płytki z pinami kontrolera. ~[OPISAĆ DOKŁADNIE KTÓRE GDZIE~~](#)

## Kod do robota

Cały, gotowy do wgrania kod do robota znajdziesz pod tym linkiem:

### [Kod do robota](#)

Pobierz plik z kodem i otwórz go we wspomnianym wcześniej Arduino IDE. Następnie podłącz robota do swojego komputera. Mając otwarty plik z kodem w IDE oraz podłączonego robota, wciśnij ikonę strzałki w programie i poczekaj aż kod zostanie wgrany



```
#include <ESP8266WiFi.h>

#ifndef APSSID
#define APSSID "Robot"
#endif
```

do robota.

A jeśli interesuje Cię co się w tym kodzie znajduje, poniżej opiszę kod krok po kroku.

Zacznijmy od dołączenia odpowiedniej biblioteki, która odpowiedzialna będzie za obsługę ESP8266.

```
#include <ESP8266WiFi.h>
```

Następnie definiujemy nazwę sieci, jeśli zmienna APSSID, w której umieszczamy nazwę naszej sieci nie jest zdefiniowana, to deklarujemy ją warunkowo.

```
#ifndef APSSID
#define APSSID "Robot"
#endif
```

I powtarzamy czynność dla zmiennej PORT.

```
#ifndef PORT
#define PORT 12345
#endif
```

Przechodzimy do zdefiniowania zmiennej odpowiedzialnej za przechowywanie serwera.

```
WiFiServer server(PORT);
const char *ssid = APSSID;
```

Na koniec definiujemy zmienne odpowiadające za piny do silników, prędkość, czas jazdy pojazdu i diodę LED.

```
int IN1=12;
int IN2=14;
int IN3=4;
int IN4=5;
int ENA=0;
int ENB=13;
```

```
int multi_=1;
int time_=250;
int led=2;
```

Musimy teraz napisać funkcje odpowiedzialne za sterowanie robotem w obu trybach, więc przejdziemy do napisania pierwszej z funkcji odpowiedzialnej za sterowanie robotem.

Funkcja będzie przyjmować argumenty odpowiedzialne za stany silników, mnożnik prędkości oraz czas poruszania.

```
void drive(int I1, int I2, int I3, int I4, int multi, int t){
```

Następnie zapisujemy ustawienia stanów silników.

```
digitalWrite(IN1,I1);
digitalWrite(IN2,I2);
digitalWrite(IN3,I3);
digitalWrite(IN4,I4);
```

I przechodzimy do napisania pętli odpowiedzialnej za płynne rozpędzanie naszego pojazdu.

```
Serial.println("Akceleracja:");
for(int i = 0; i < 1024; i += multi) {
```

Ustawiamy prędkości silników.

```
analogWrite(ENA,i);
analogWrite(ENB,i);
Serial.print(i);
Serial.print("...");
delay(1);
}
```

Skoro już rozpędziliśmy naszego robota, musimy teraz napisać pętlę odpowiedzialną za jego płynne hamowanie.

```
delay(t);
Serial.println("Hamowanie:");
for(int i = 1024; i > 0; i -= multi) {
```

Tak jak przy rozpędzaniu, ustawiamy prędkości silników również przy hamowaniu i kończymy jedną z funkcji odpowiedzialną za sterowanie.

```
analogWrite(ENA,i);
analogWrite(ENB,i);
Serial.print(i);
Serial.print("...");
delay(1);
}
```

Przejdźmy teraz do napisania drugiej z funkcji, tym razem odpowiedzialnej za swobodne sterowanie pojazdem. Funkcja będzie przyjmować argumenty odpowiedzialne za stany silników.

```
void freeDrive(int I1, int I2, int I3, int I4) {
```

Ustawiamy prędkość na 100%, tak aby mieć jak najwięcej zabawy podczas korzystania z pojazdu w danym trybie.

```
analogWrite(ENA,1024);
analogWrite(ENB,1024);
```

Zapisujemy ustawienia stanów silników i zamkamy drugą z funkcji odpowiedzialnych za sterowanie.

```
digitalWrite(IN1,I1);
digitalWrite(IN2,I2);
digitalWrite(IN3,I4);
```

```
digitalWrite(IN4, I4);  
}
```

Następnie piszemy małą funkcję odpowiedzialną za zapisywanie ustawień użytkownika dotyczących prędkości oraz czasu poruszania pojazdu.

```
void setVars(int multi, int t){  
    multi_ = multi;  
    time_ = t;  
}
```

Potrzebna będzie jeszcze funkcja z ustawieniami pinów oraz wifi.

```
void setup() {
```

Ustawiamy wcześniej zdefiniowanym pinom tryb wyjścia.

```
pinMode(IN1, OUTPUT);  
pinMode(IN2, OUTPUT);  
pinMode(IN3, OUTPUT);  
pinMode(IN4, OUTPUT);  
pinMode(ENA, OUTPUT);  
pinMode(ENB, OUTPUT);
```

Następnie ustawiamy tryb WiFi jako tryb dostępu (Access Point) i dla pewności wykonujemy rozłączenie z siecią. Tak na wypadek gdyby został wykonany soft reset.

```
WiFi.mode(WIFI_STA);  
WiFi.disconnect();  
delay(100);
```

Rozpoczynamy sesje dla serial monitora.

```
Serial.begin(115200);  
Serial.println();  
Serial.print("Konfiguracja AP...");
```

Ustawiamy adres ip, bramę domyślną oraz maskę robota i następnie zapisujemy konfigurację sieci. Warto tutaj dodać, że oktety w programie dla adresu ip, bramy oraz maski są oddzielone przecinkiem, a nie kropką tak jak się domyślnie przyjęło.

```
IPAddress ip(192, 168, 1, 200);  
IPAddress gateway(192, 168, 1, 254);  
IPAddress subnet(255, 255, 255, 0);  
WiFi.softAPConfig(ip, gateway, subnet);
```

Następnie włączamy tryb dostępu (Access Point)

```
WiFi.softAP(ssid);  
IPAddress myIP = WiFi.softAPIP();
```

oraz pobieramy adres IP i wyświetlamy go w monitorze.

```
Serial.print("AP IP: ");  
Serial.println(myIP);
```

Rozpoczynamy sesje serwera TCP.

```
server.begin();
```

Na końcu ustawień dla pewności zatrzymujemy silniki i kończymy funkcję z ustawieniami.

```
digitalWrite(IN1, LOW);  
digitalWrite(IN2, LOW);  
digitalWrite(IN3, LOW);  
digitalWrite(IN4, LOW);  
analogWrite(ENA, 0);  
analogWrite(ENB, 0);  
}
```

Została ostatnia z funkcji. Mianowicie funkcja odpowiedzialna za sprawdzanie czy klient (w tym wypadku mata) jest podłączony, odczytywanie komend a następnie ich wykonanie.

```
void loop() {
```

Na początku zapisujemy informacje o połączonym kliencie.

```
WiFiClient client = server.available();
```

Następnie sprawdzamy warunkowo czy zmienna z klientem istnieje oraz czy sam klient jest już połączony.

```
if (client) {
    if(client.connected()) {
        Serial.println("Client Connected");
    }
}
```

Tworzymy pętle, która będzie się wykonywać, jeśli klient jest połączony z robotem oraz dopóki liczba połączonych użytkowników będzie większa niż zero.

```
while(client.connected()) {
    while(client.available()>0) {
```

Teraz trzeba odczytać sekwencję znaków przesyłanych przez użytkownika (matę).

Sekwencja ta kończy się będzie znakiem nowej linii.

```
String line = client.readStringUntil('\n');
```

Następnie tworzymy tablice znaków (char), konwertujemy wcześniej odczytaną sekwencję do pojedynczych znaków i zapisujemy je w utworzonej tablicy znaków.

```
char cmd[line.length()+1];
line.toCharArray(cmd, line.length()+1);
```

Przechodzimy do pętli, która będzie wykonywała komendy znajdujące się w tablicy.

```
for(int i = 0; i < line.length()+1; i++) {
```

Teraz musimy sprawdzać komendy switchem i następnie wykonać je odpowiednimi case'ami. Każdy przypadek (case) zawiera funkcję "drive" lub "freeDrive", która zostanie wykonana z argumentami dla konkretnej komendy.

```
switch(cmd[i]) {
    case 'u':
        drive(HIGH, LOW, HIGH, LOW, multi_, time_);
        break;
    case 'd':
        drive(LOW, HIGH, LOW, HIGH, multi_, time_);
        break;
    case 'r':
        drive(LOW, HIGH, HIGH, LOW, multi_, time_);
        break;
    case 'l':
        drive(HIGH, LOW, LOW, HIGH, multi_, time_);
        break;
    case 'z':
        freeDrive(HIGH, LOW, HIGH, LOW);
        break;
    case 'x':
        freeDrive(LOW, HIGH, LOW, HIGH);
        break;
    case 'c':
        freeDrive(LOW, HIGH, HIGH, LOW);
        break;
    case 'v':
        freeDrive(HIGH, LOW, LOW, HIGH);
        break;
}
```

Na końcu switcha dodajemy jeszcze przypadek, w którym użytkownik podaje nieznane komendy i wykonujemy w niej funkcję drive z argumentami zatrzymującymi pojazd.

```
default:  
    drive(LOW, LOW, LOW, LOW, 1, 0);  
    break;  
}  
}
```

Odczytujemy jeszcze komendy podane przez serial monitor, gdy aktywna jest jego sesja.

```
while(Serial.available()>0){  
    client.write(Serial.read());  
}  
}
```

Na koniec zatrzymujemy połączenie z użytkownikiem.

```
client.stop();  
Serial.println("Client disconnected");  
}
```

## 4. Mata sterująca

( tu wstawić zdjęcie maty )

**Lista wymaganych materiałów:**

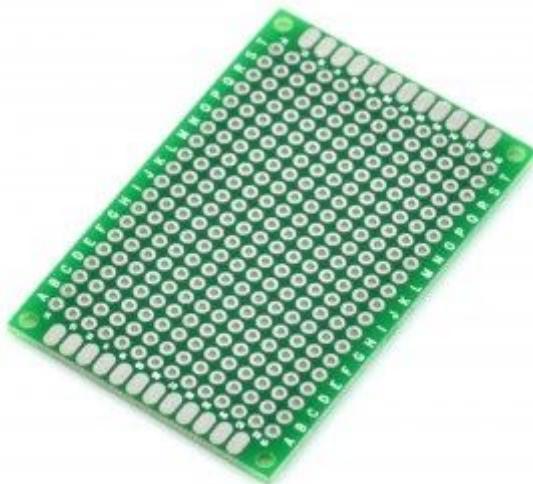
- 1) **Raspberry Pi Zero W 512MB RAM - WiFi + BT 4.1 - 1 sztuka**



**Przykładowy link do sklepu:**

[https://botland.com.pl/pl/moduly-i-zestawy-raspberry-pi-zero/8330-raspberry-pi-zero-w-512mb-ram-wifi-bt-41.html?search\\_query=pi+zero&results=191](https://botland.com.pl/pl/moduly-i-zestawy-raspberry-pi-zero/8330-raspberry-pi-zero-w-512mb-ram-wifi-bt-41.html?search_query=pi+zero&results=191)  
<https://botland.com.pl/pl/gniazda-szpilkowe-goldpin/12659-wtyk-goldpin-2x20-prosty-raster-254mm.html>

**2) Płytki uniwersalna:**



**Przykładowy link do sklepu:**

[9cm x 15cm](#)

[2cm x 8cm](#)

Wymiary minimalne: 9cm x 15cm

Komentarz: + dodatkowa 2cm x 8cm do budowy płytki zasilającej

**3) Części do płytki drukowanej:**

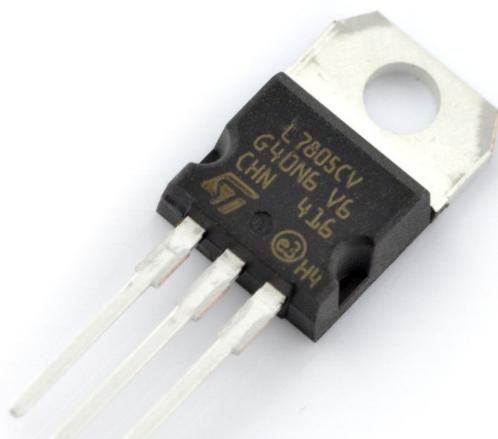
**Przykładowy link do sklepu:**

<https://botland.com.pl/pl/laminaty/1054-laminat-fr4-dwustronny-rozmiar-l.html>

<https://botland.com.pl/pl/laminaty/1943-laminat-dwustronny-150x210mm.html>

[https://botland.com.pl/pl/wytrawiacze/1057-wytrawiacz-b327-100g-5901764329183.html?search\\_query=wytrawiacz&results=7](https://botland.com.pl/pl/wytrawiacze/1057-wytrawiacz-b327-100g-5901764329183.html?search_query=wytrawiacz&results=7)

**4) Zasilanie do RPi - L7805CV - 1 sztuka.**



**Przykładowy link do sklepu:**

<https://abc-rc.pl/product-pol-6839-Elektronika-Stabilizator-L7805CV-5V-1-5A-obudowa-TO-220.html>

- Napięcie wyjściowe: 5V
- Maksymalne napięcie wejściowe: 35V
- Prąd wyjściowy: 1,5A
- Liczba pinów: 3
- Obudowa: TO-220
- Nr katalogowy: SNL7805CV

**5) Baterie - Model: "18650" - 1 sztuka.**



[https://botland.com.pl/pl/koszyki-na-baterie/5241-koszyk-na-3-baterie-typu-18650.html?search\\_query=koszyk+18650&results=5](https://botland.com.pl/pl/koszyki-na-baterie/5241-koszyk-na-3-baterie-typu-18650.html?search_query=koszyk+18650&results=5)

<https://www.gotronik.pl/akumulator-li-ion-18650-ogniwo-litowo-jonowe-3400-mah-panasonic-ncr-18650b-p-6931.html>

(same baterie można dużo taniej znaleźć na Allegro lub OLX)

**6) Wtyk goldpin 2x40 prosty raster 2,54mm - 1 raster powinien wystarczyć.**



**Przykładowy link do sklepu:**

[https://botland.com.pl/pl/gniazda-szpilkowe-goldpin/204-wtyk-goldpin-2x40-prosty-raster-254mm.html?results=76&search\\_query=goldpin](https://botland.com.pl/pl/gniazda-szpilkowe-goldpin/204-wtyk-goldpin-2x40-prosty-raster-254mm.html?results=76&search_query=goldpin)

**7) Tact Switch 12x12mm z nasadką - 18 sztuk**



**Przykładowy link do sklepu:**

[https://botland.com.pl/pl/tact-switch/11138-tact-switch-12x12mm-z-nasadka-kwadrat-czarny-5szt.html?search\\_query=przycisk+tact&results=45](https://botland.com.pl/pl/tact-switch/11138-tact-switch-12x12mm-z-nasadka-kwadrat-czarny-5szt.html?search_query=przycisk+tact&results=45)

**8) Wyświetlacz OLED niebieski graficzny 1,3" 128x64px I2C v2 - 1 sztuka.**



- Napięcie pracy: od 3,3 V do 5,0 V
- Sterownik: SH1106 (dokumentacja)
- Komunikacja: I2C

- Typ wyświetlacza: OLED
- Przekątna: 1,3"
- Rozdzielcość: 128 x 64 px
- Kolor znaków: biały
- Kąt widzenia: powyżej 160 °
- Temperatura pracy: od -20 °C do 70 °C
- Wymiary: 35 x 33 mm

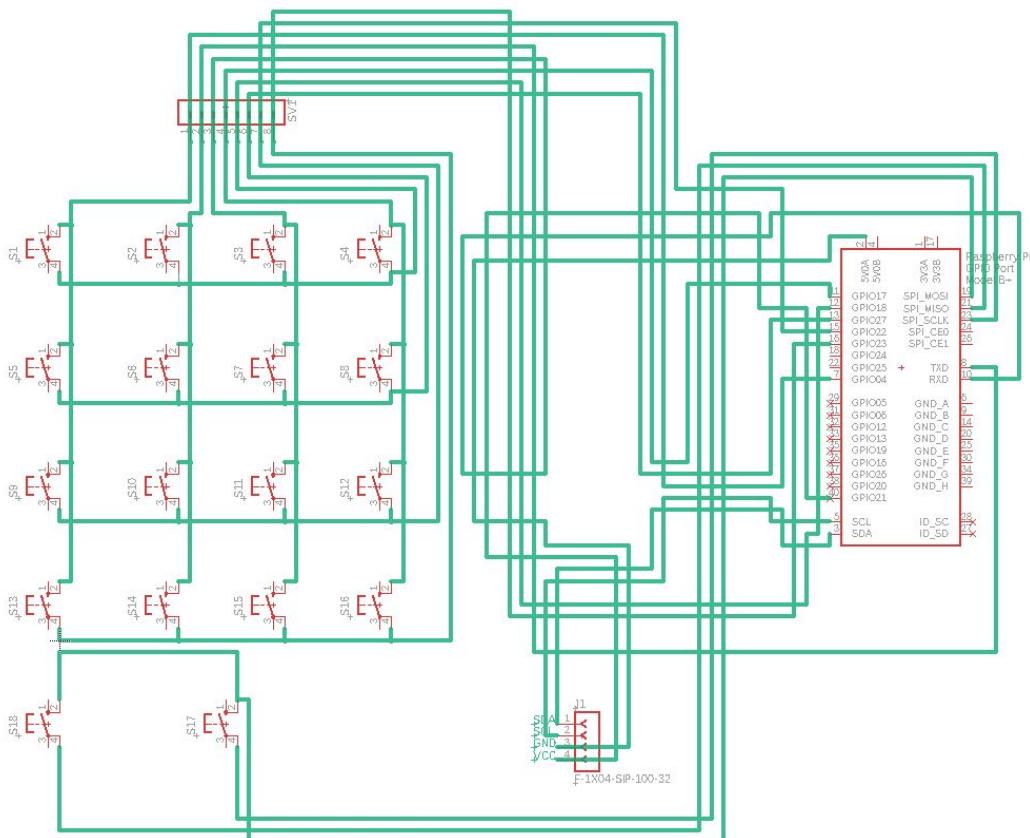
**Przykładowy link do sklepu:**

<https://botland.com.pl/pl/wyswietlacz-oled/8246-wyswietlacz-oled-niebieski-graficzny-13-128x64px-i2c-v2-niebieskie-znaki.html>

## Budowa maty

Na wstępnie chciałbym zaznaczyć, że uniwersalne płytki PCB dają nam ogromną swobodę w tym jak nasz efekt końcowy będzie się prezentować.

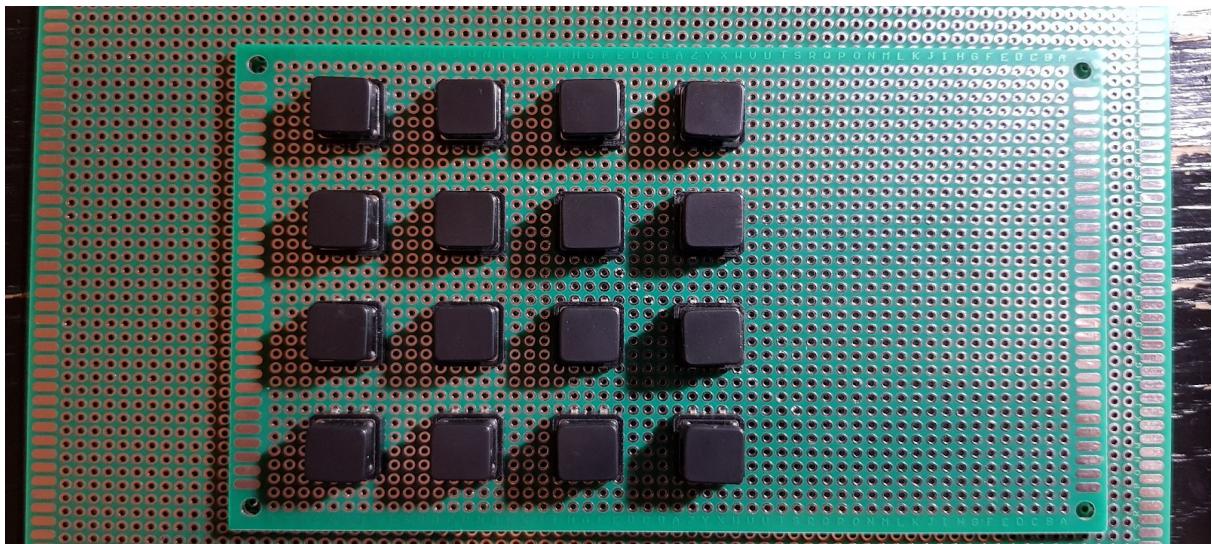
Przed rozpoczęciem budowy "maty" najlepiej zapoznać się z schematem i wstępnie rozplanować sobie rozkład połączeń na uniwersalnej płytce. Porządek to podstawa,



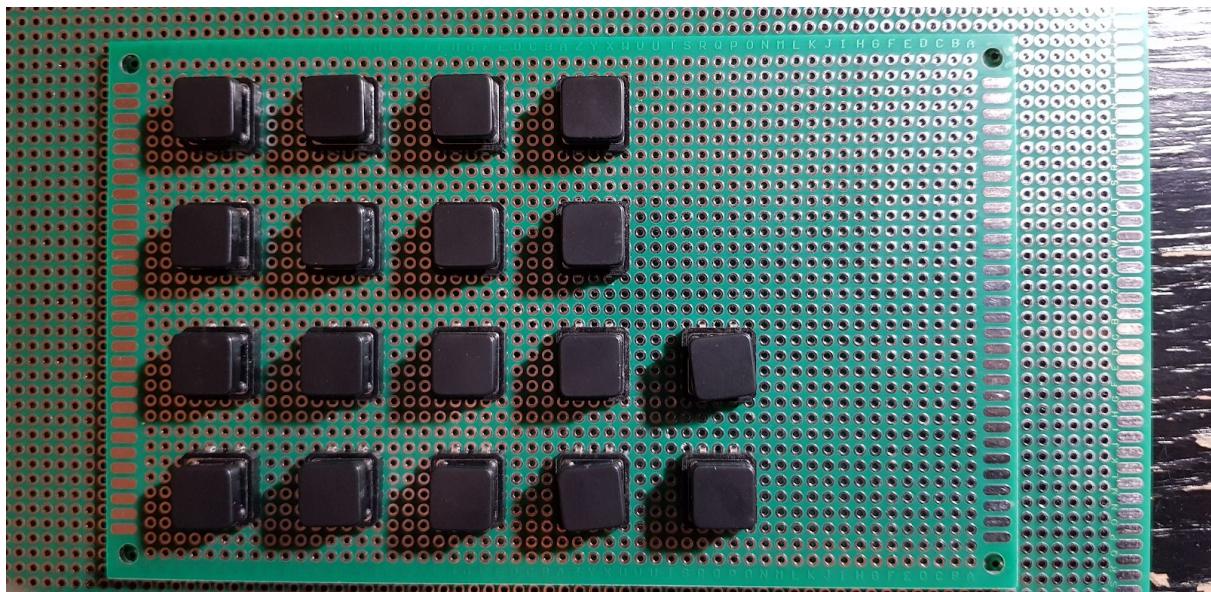
ponieważ później pozwoli nam to na łatwe modyfikacje i naprawy ewentualnych błędów.

Pierwszym krokiem będzie rozkład przycisków S1 - S16.

Z racji tego, że wybrana płytka jest dosyć dużych rozmiarów, bo ma aż 9cm na 15cm, to możemy pozwolić sobie na spore odstępy między przyciskami tak jak na poniższym zdjęciu (oczywiście każdy może mieć większe bądź mniejsze odstępy, to już według osobistych preferencji):

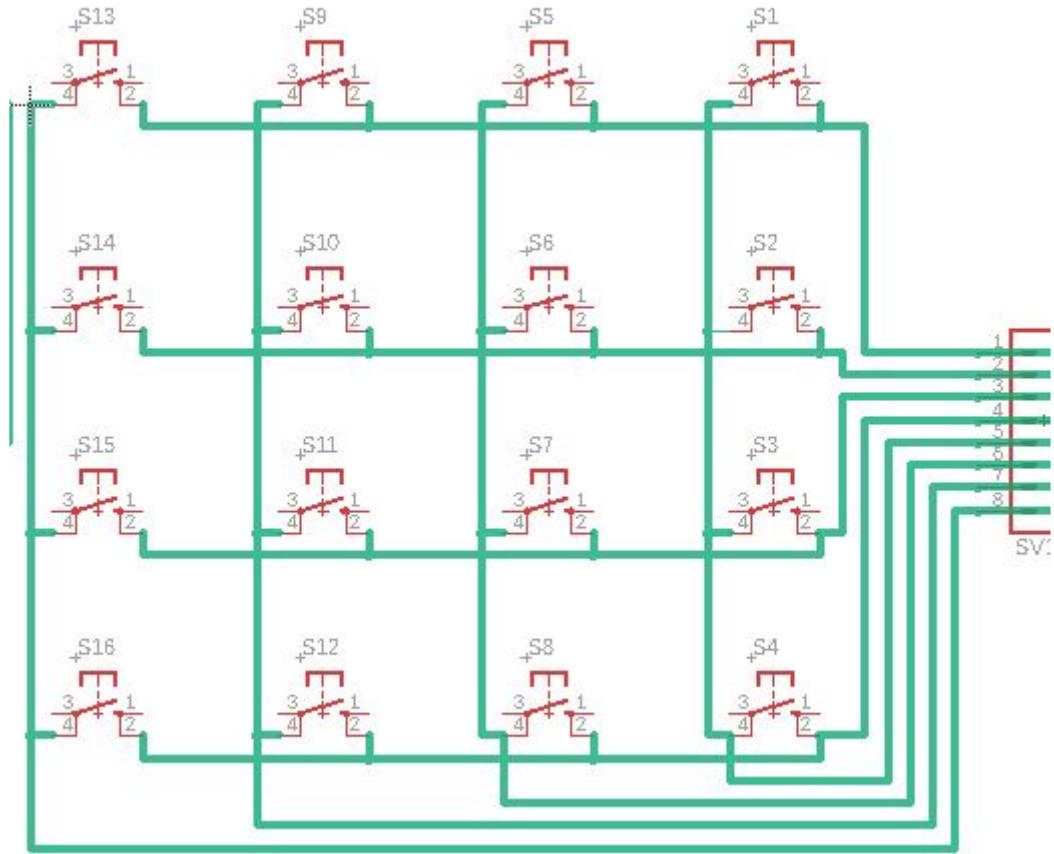


Następnie możemy dodać kolejne dwa przyciski (S17 i S18), które będą przyciskami funkcyjnymi:

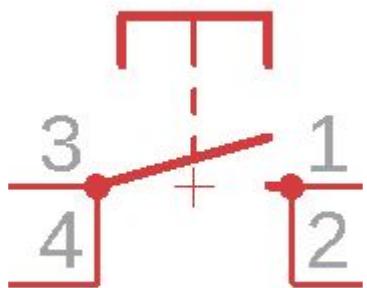


Teraz gdy już mamy rozplanowane rozłożenie przycisków możemy przejść do przypomnienia sobie schematu samej klawiatury jak i samego przycisku co ułatwi nam lutowanie.

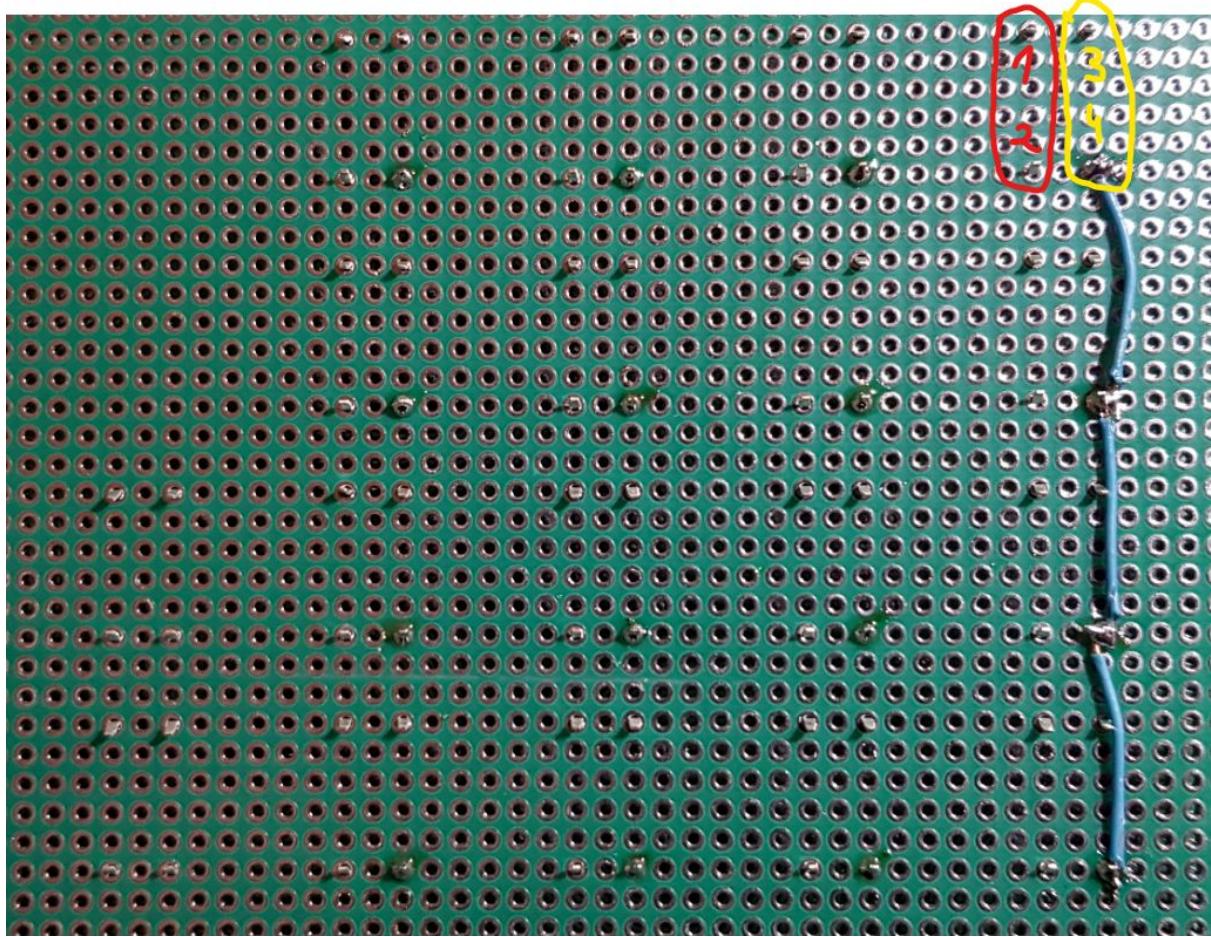
- Klawiatura:



- Przycisk:

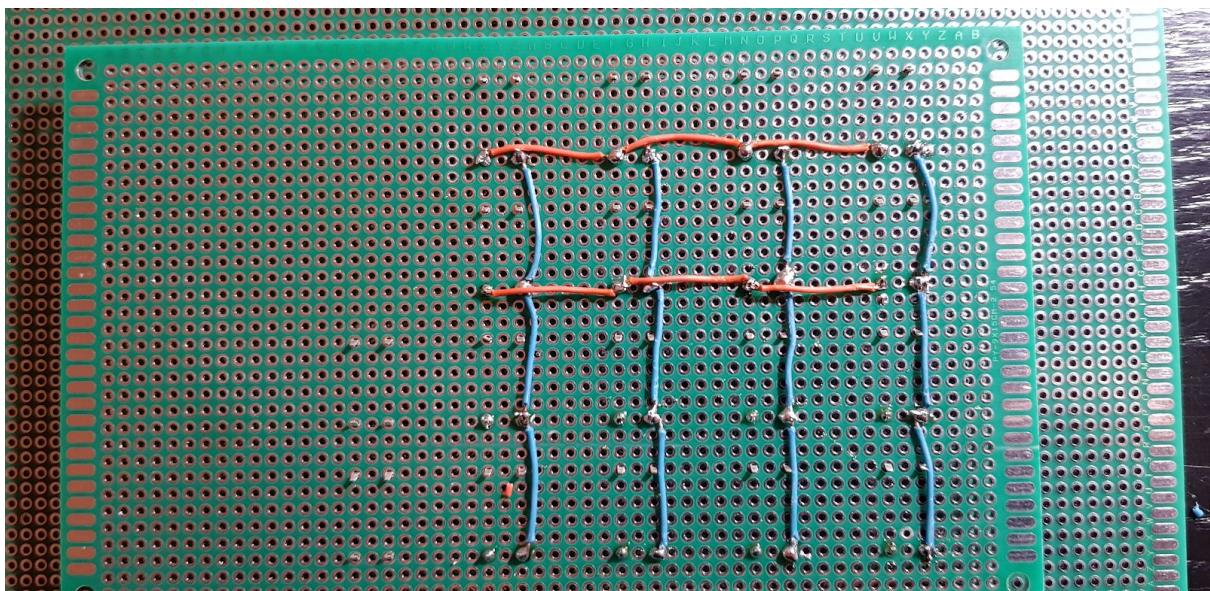
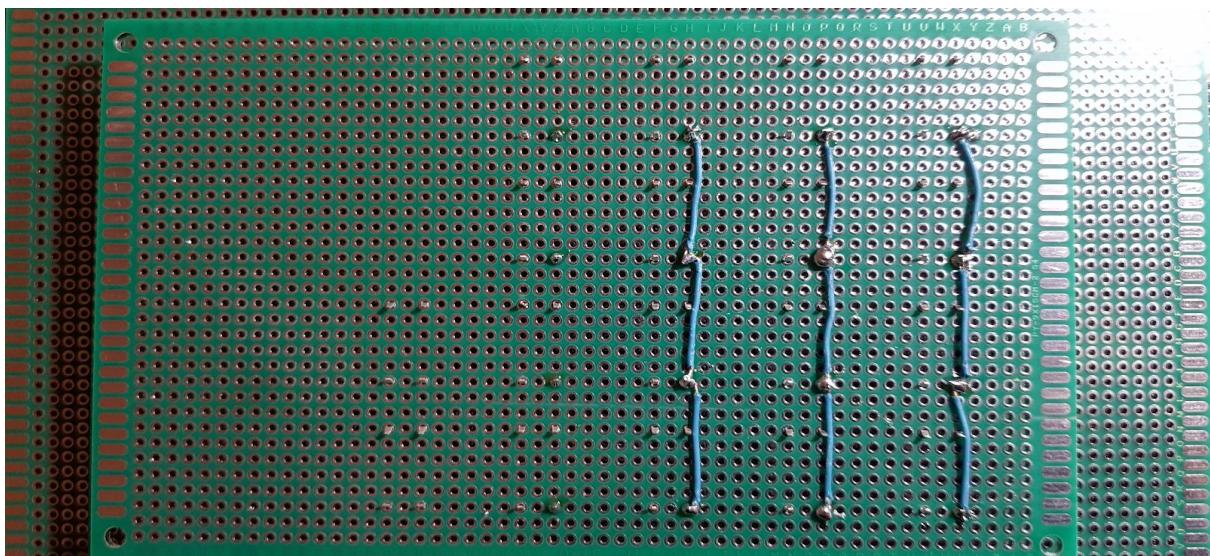
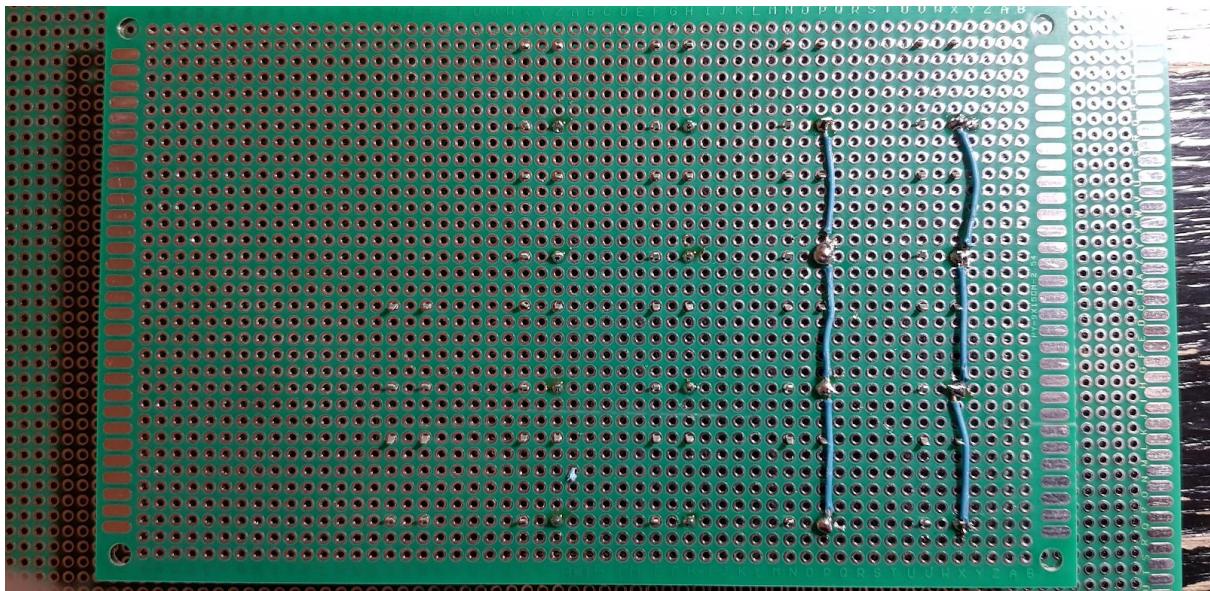


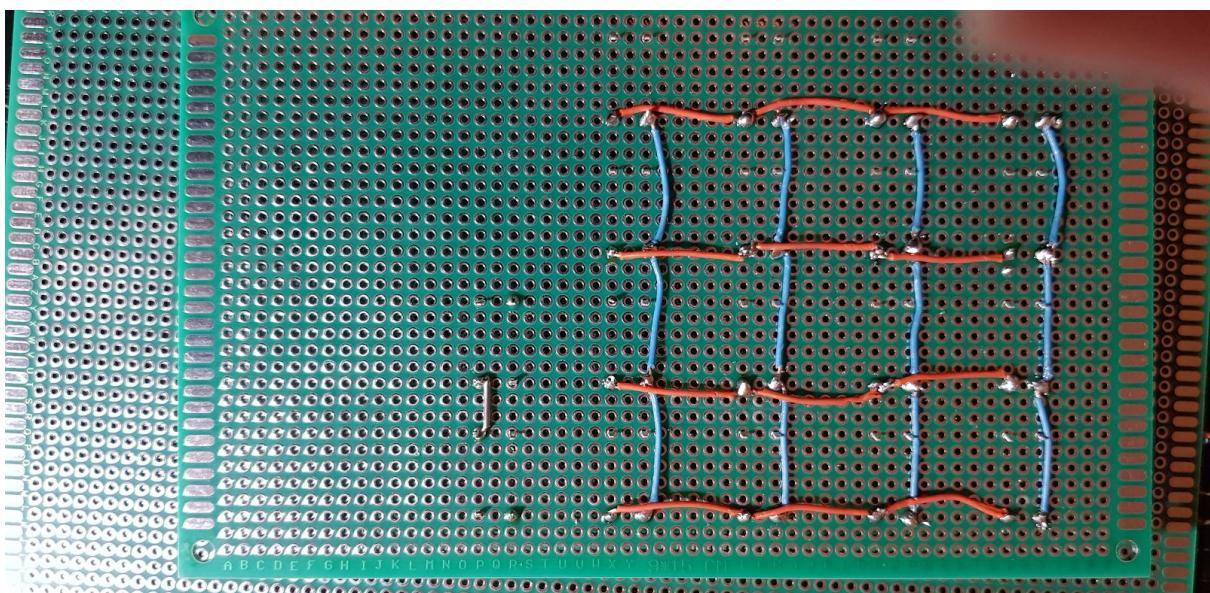
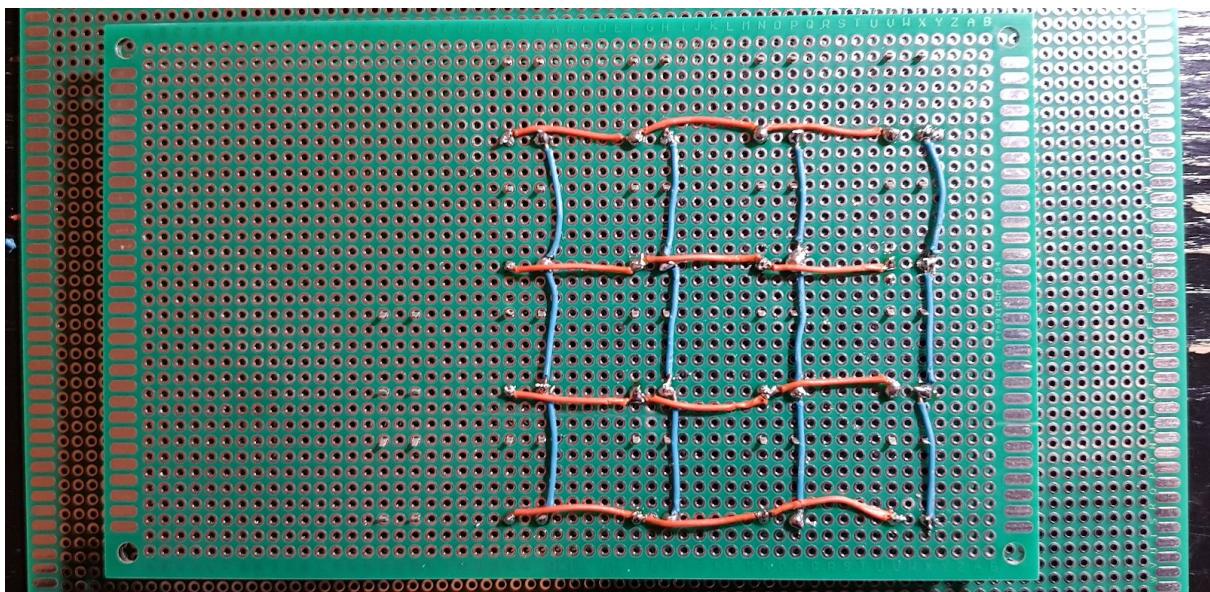
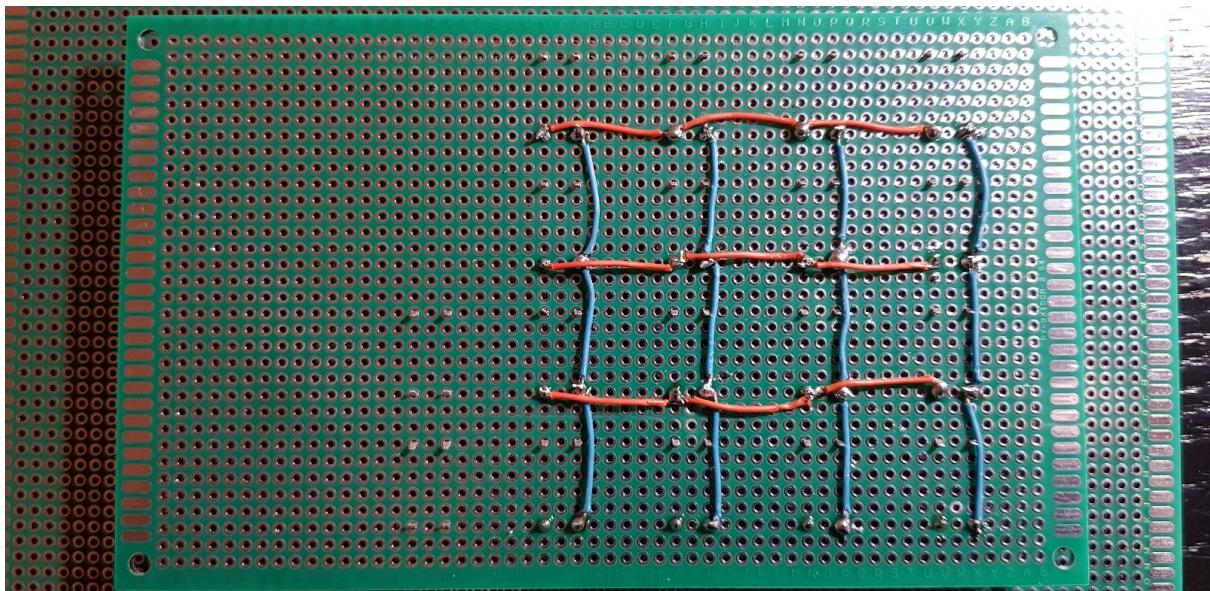
Podstawową rzeczą jaką powinniśmy zauważać patrząc na schemat przycisku jest fakt, że nóżki 3 i 4 oraz 1 i 2 są ze sobą połączone, więc wciśnięcie przycisku powoduje zamknięcie obwodu między parami nóżek 1,2 i 3,4. Jest to o tyle istotny fakt, że podczas lutowania możemy zaoszczędzić bardzo dużo miejsca jak i czasu poświęconego na pracę. Przejdźmy zatem do lutowania:



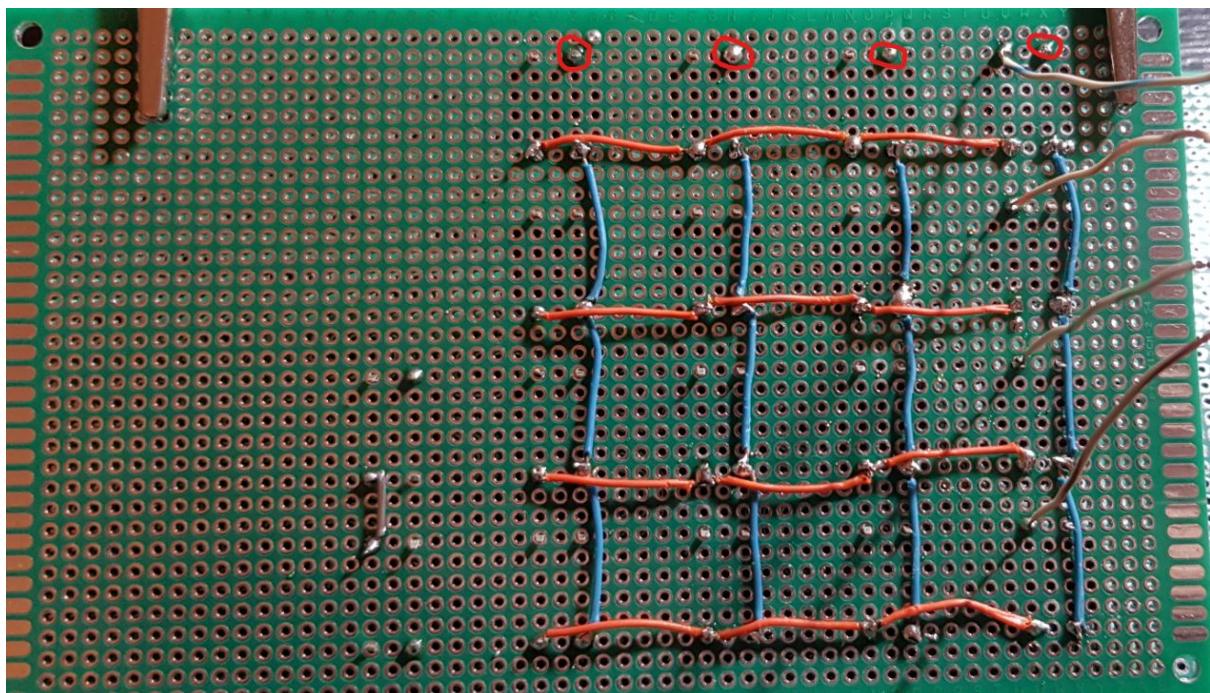
Na zdjęciu powyżej widzimy pierwszą zlutowaną kolumnę. Oczywiście można było zrobić to inaczej, ale tak jak wspominałem, na płytach uniwersalnych mamy pełną dowolność.

Poniżej zdjęcia będą przedstawiać postęp w lutowaniu kolumn i wierszy między przyciskami.

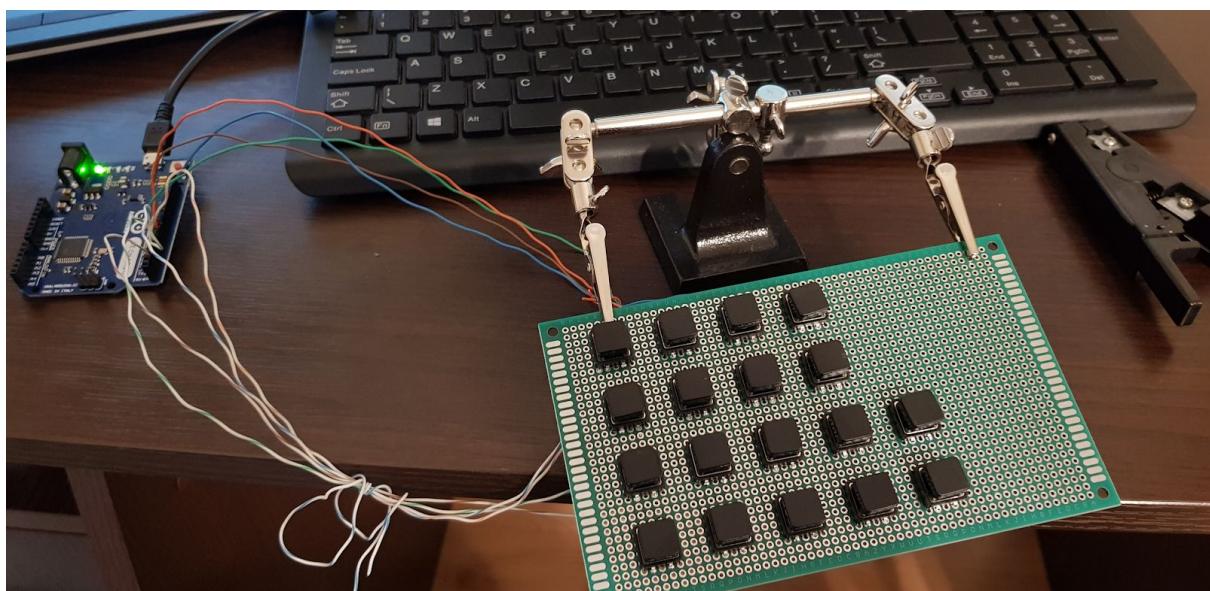




Następnie aby sprawdzić czy wszystko zostało podłączone tak jak należy oraz udało nam się uniknąć "zimnych lutów" możemy na szybko podłączyć klawiaturę do np. Arduino i sprawdzić czy działa poprawnie:



(Na czerwono są zaznaczone miejsca w których zostały przylutowane kable od kolumn, a nie ma ich na zdjęciu)



Powyżej widać klawiaturę podpiętą do Arduino,

a poniżej widać program oraz wynik na konsoli po naciśnięciu klawiszy.

The screenshot shows the Arduino IDE interface. On the left, the code for 'Klaw' is displayed:

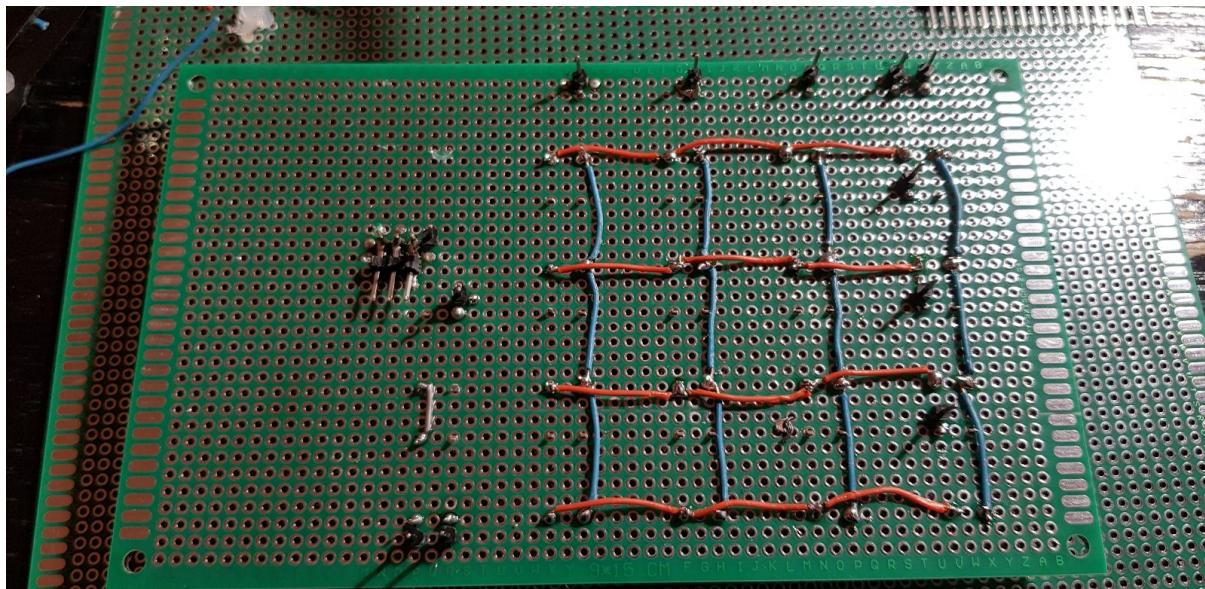
```
#include <Keypad.h> //biblioteka od klawiatury
const byte ROWS = 4; // ile wierszy
const byte COLS = 4; //ile kolumn
byte rowPins[ROWS] = {5, 4, 3, 2}; //piny wierszy
byte colPins[COLS] = {6, 7, 8, 9}; //piny kolumn
char keys[ROWS][COLS] = { //mapowanie klawiatury
    {'1','2','3','A'},
    {'4','5','6','B'},
    {'7','8','9','C'},
    {'*',0,'#','D'}
};
Keypad klawiatura = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS ); //inicjalizacja klawiatury
void setup(){
    Serial.begin(9600);
}
void loop(){
    char klawisz = klawiatura.getKey();
    if (klawisz){
        Serial.println(klawisz);
    }
}
```

On the right, the terminal window titled 'COM3' shows the output of the program. It displays the characters '1', '2', '3', 'A', '4', '5', '6', 'B', '7', '8', '9', 'C', '\*', '0', '#', and 'D' sequentially, each followed by a new line. There are also some status indicators at the bottom of the terminal window.

(Kod do Arduino zostanie udostępniony na GitHubie jako plik [test\\_klawiatury\\_4x4](#))

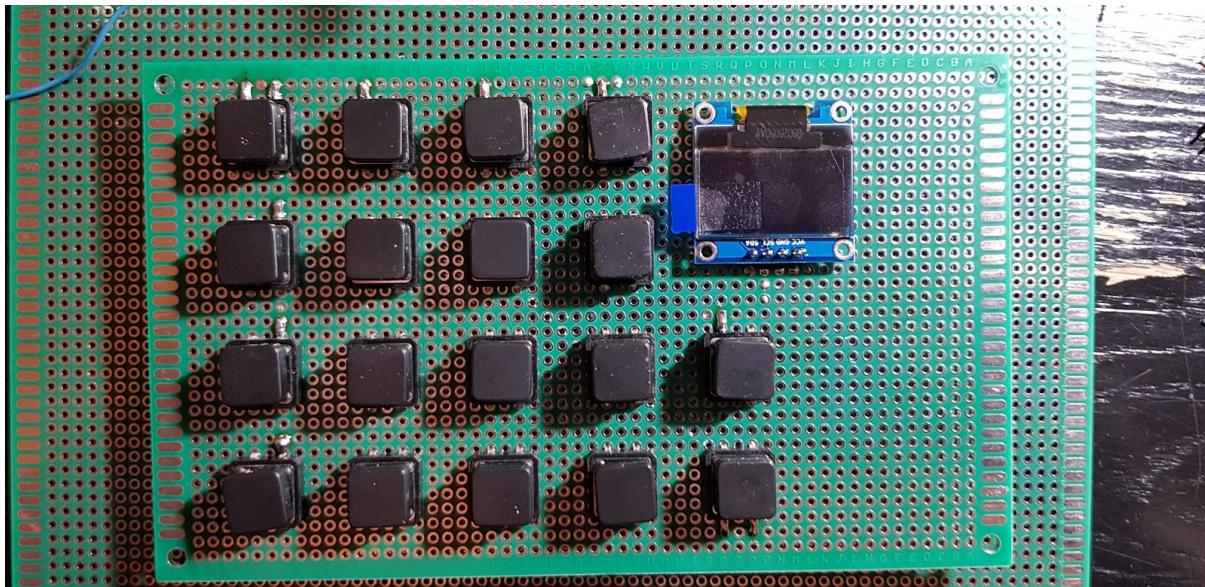
Po poprawnych wynikach otrzymanych w teście możemy przejść do dodania pinów do klawiatury aby później można było ją podłączyć do RPi Zero.

(Nie zastosowałem tutaj kabli przylutowanych na stałe, ponieważ nie miałem odpowiednich pod ręką, a kable ze skrętki nie były odpowiednie, ponieważ szybko się łamały i były zbyt sztywne)



Powyżej można zauważyć 4 dodatkowe piny obok przycisków funkcyjnych, ponieważ w międzyczasie został przylutowany ekran OLED co będzie widoczne na następnym

zdjęciu:



Po skończonej klawiaturze możemy przejść do zasilania jakiego użyjemy dla naszego Raspberry Pi Zero.

Ważne jest to aby pamiętać o podstawowych parametrach jakie trzeba zapewnić RPi aby działało stabilnie. Po pierwsze najważniejsze jest napięcie, jeśli chcemy zasilić RPi poprzez złącze microUSB to musimy zapewnić następujące wartości:

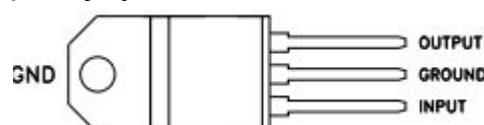
- Napięcie [V]: 4.75 - 5.25
- Natężenie [mA]: min. 500

Jest również możliwość zasilania RPi poprzez GPIO, ale jest to dosyć ryzykowny zabieg.

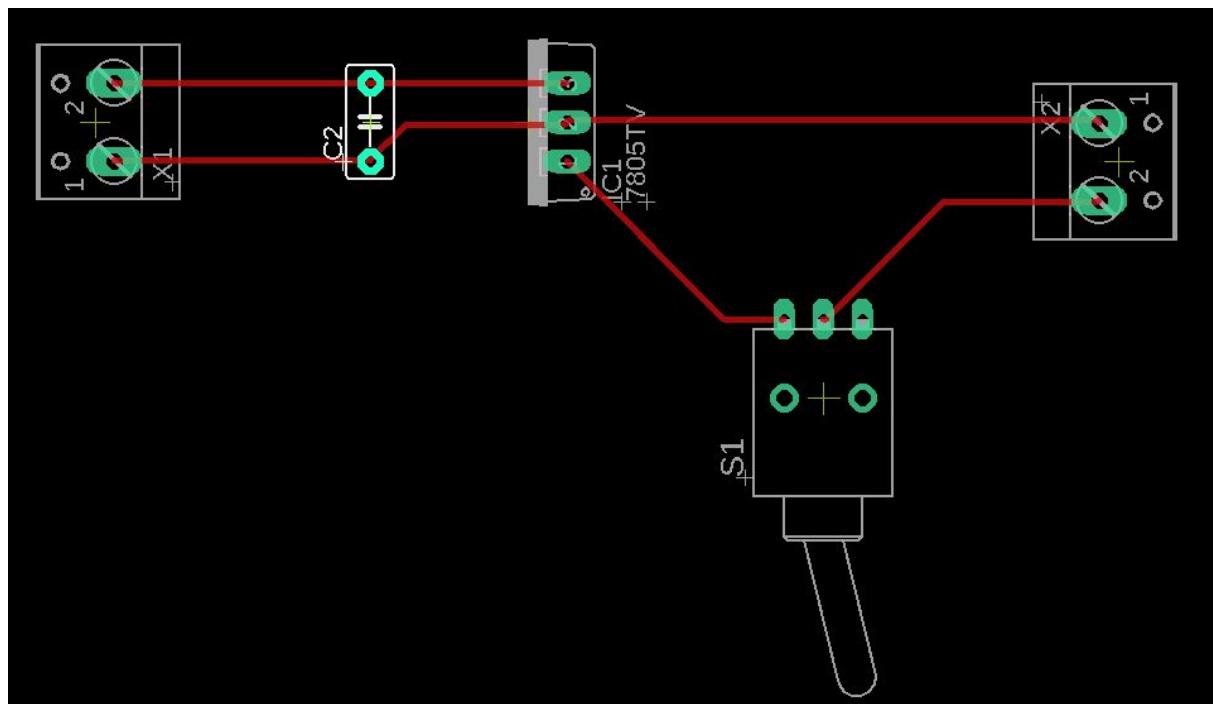
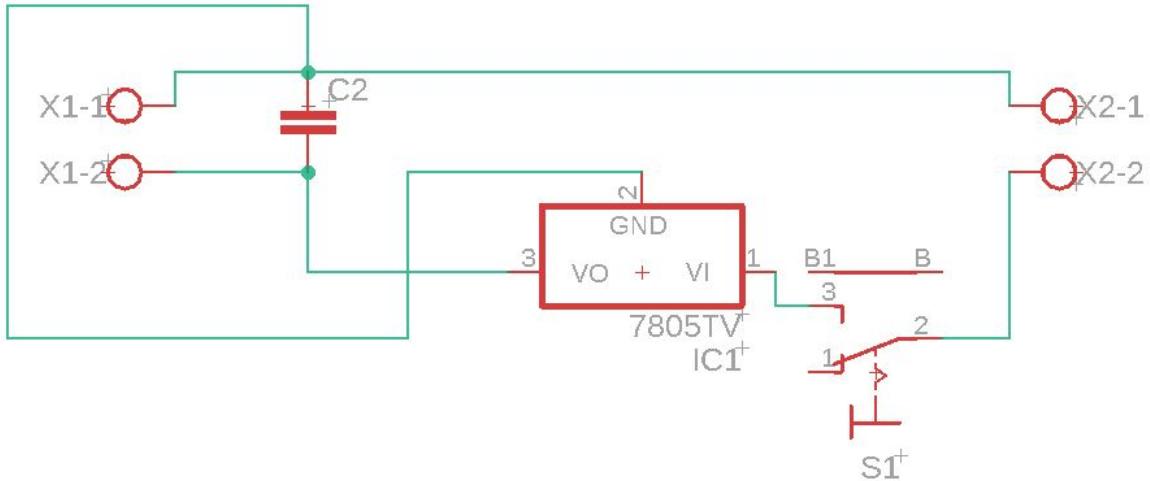
Do wykonania prostego zasilacza będziemy potrzebowali:

- L7805CV
- Koszyk na baterie 3x 18650
- kawałek płytki uniwersalnej
- trochę kabla
- przełącznik
- kondensator (ja wybrałem 10v 470uF)
- oraz opcjonalnie można użyć złącz ARK

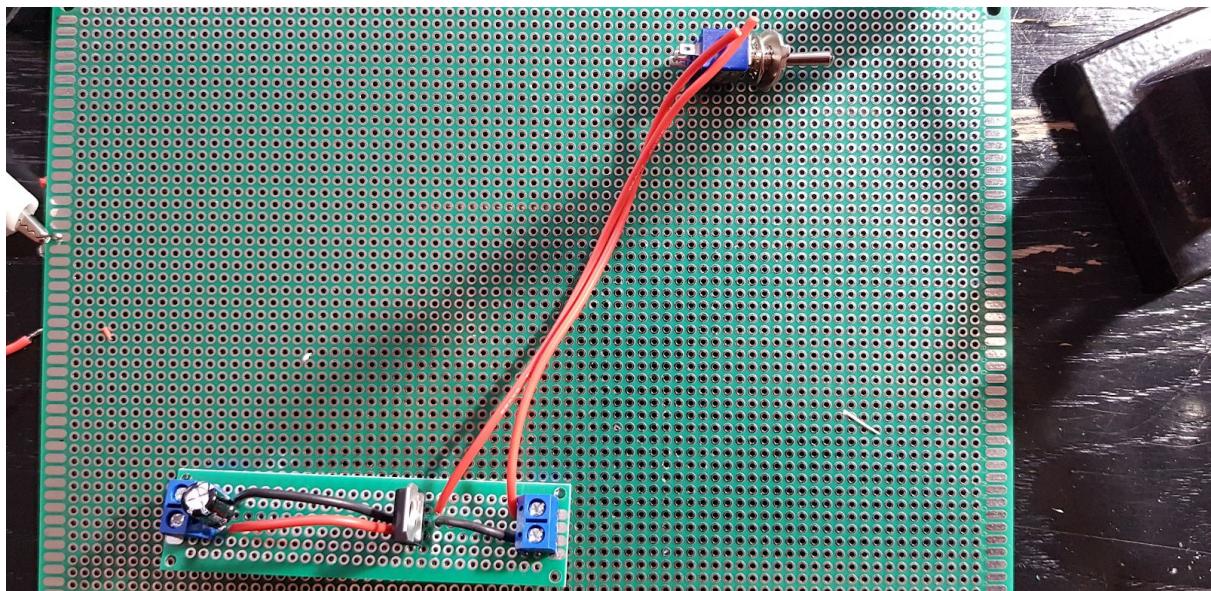
Na wstępie mały obrazek z wyjściami dla L7805CV aby wszystko poprawnie podłączyć:



Następnie schemat całego zasilacza:

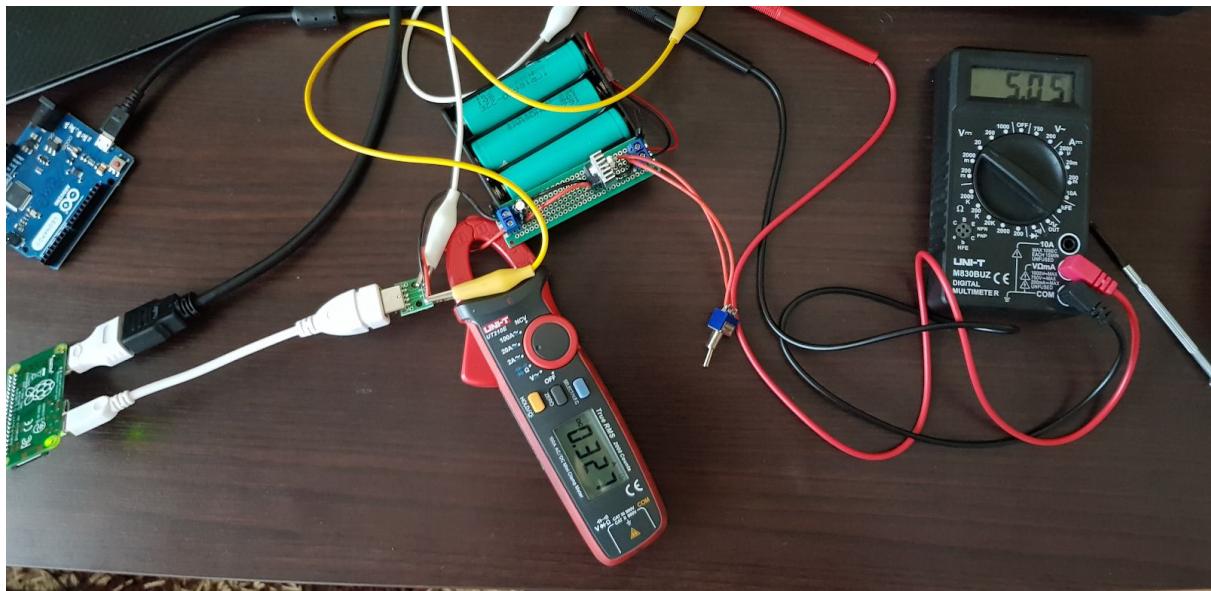


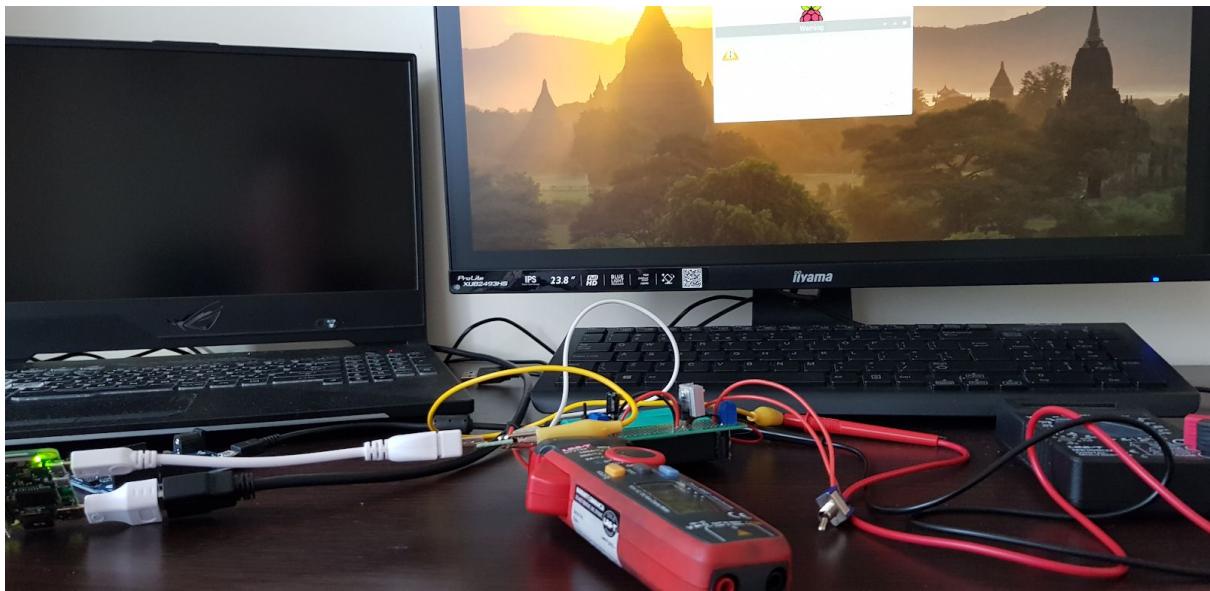
I końcowy wygląd:



Kolejna część praktycznie skończona, więc możemy przejść do kolejnych testów z udziałem Raspberry Pi Zero i naszego zasilacza.

Na wstępie dodam jeszcze, żeby połączyć zasilacz z Raspberry użyłem starej końcówki microUSB z ładowarki która miała jedynie 2 żyły (czerwoną i czarną) przez co łatwiej było podłączyć ją pod zasilacz.





Na powyższych zdjęciach widzimy pakiet 3 baterii 18650, dwa multymetry z czego jeden mierzy natężenie między zasilaczem a Raspberry (w stanie spoczynku przy uruchomionym systemie Raspbiana oraz podłączonym ekranie poprzez HDMI pobór wynosił około 290-340mA.)

Drugi multymetr (czarny) pokazuje napięcie na wyjściu zasilacza, które u mnie wynosiło 5,04 - 5,05V nawet przy większym obciążeniu RPi. Pakiet baterii na wyjściu miał napięcie w przedziale 12 - 12,10V.

## Obudowa maty

Teraz możemy zostawić elektronikę na boku i przejść do budowy.

Ja użyłem do tego plexy, ale równie dobrze możemy stworzyć obudowę z drewna lub zaprojektować i wydrukować w drukarce 3D.

Wymiary obudowy prezentowanej na zdjęciach mają:

- Wysokość: 70mm
- Szerokość: 100mm
- Długość: 160mm

Oczywiście szerokość i długość można dopasować idealnie do płytki uniwersalnej i zastosować szerokość 90mm i długość 150mm.

Zdjęć z samego wycinania nie będzie lecz przedstawię swoje osobiste doświadczenia jakie uzyskałem podczas tego procesu.

Wskazówki:

- Do budowy obudowy dobrze jest wykorzystać plexi o grubości 2mm, ponieważ w łatwy sposób możemy ciąć ją nożem do tapet. Po 3-4 mocniejszych nacięciach i mocniejszym nagięciu plexi powinna przełamać się.
- Dobrze jest też nakleić sobie taśmę papierową i na niej narysować linię gdzie będziemy cięli, ponieważ plastik jest dosyć ślizgi i możemy zrobić sobie krzywdę.
- Możemy również ciąć wzdłuż kątownika lub innej prostej rzeczy aby nóż nie uciekał na boki. tak aby nie było większych nierówności.
- Po cięciu brzegi plexi dobrze jest wyrównać papierem ściernym. Ja użyłem 120 i 60
- Ostatnią wskazówką jest to aby nie ciąć plexi wyrzynarką, przynajmniej mi się to nie udało, ponieważ brzeszczot podczas cięcia uzyskuje wysoką temperaturę i topi plexi.



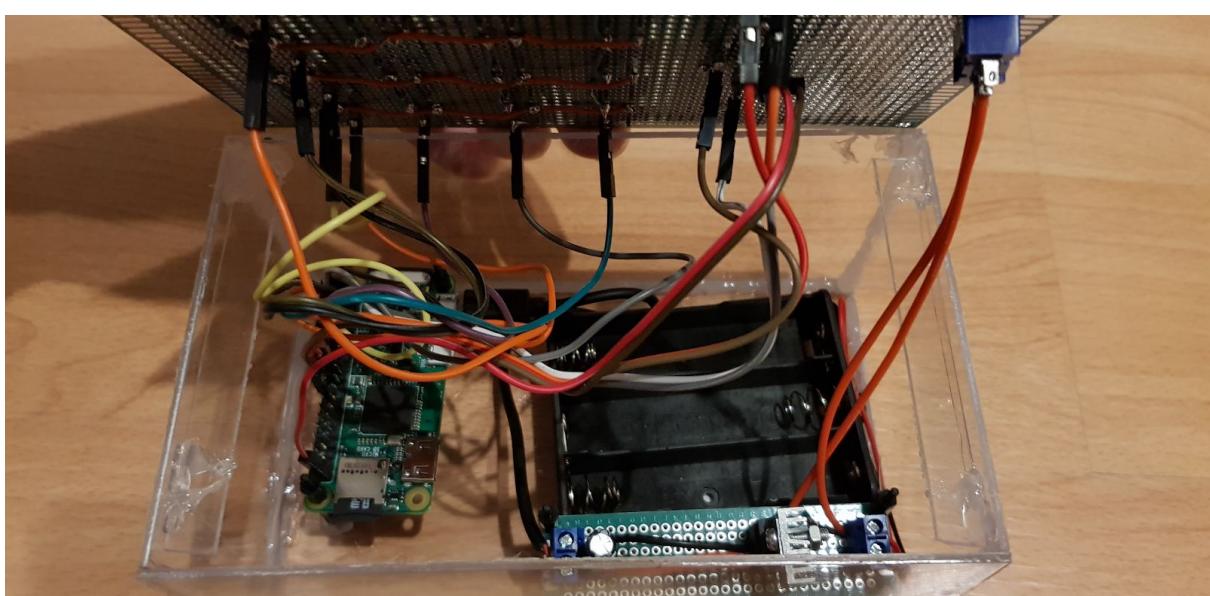
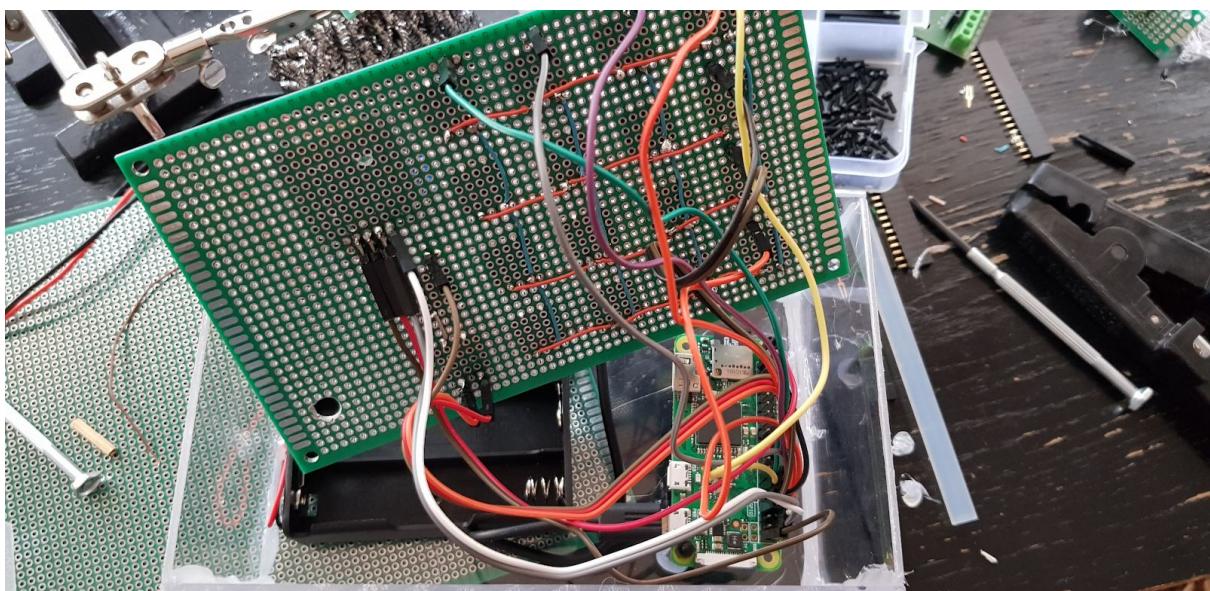
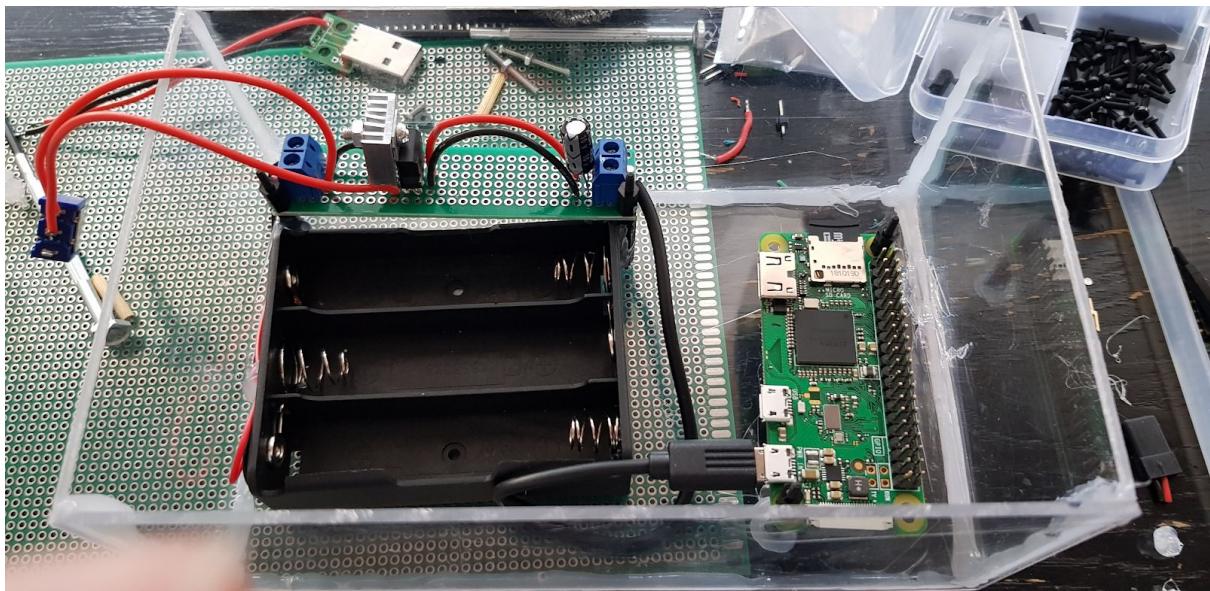


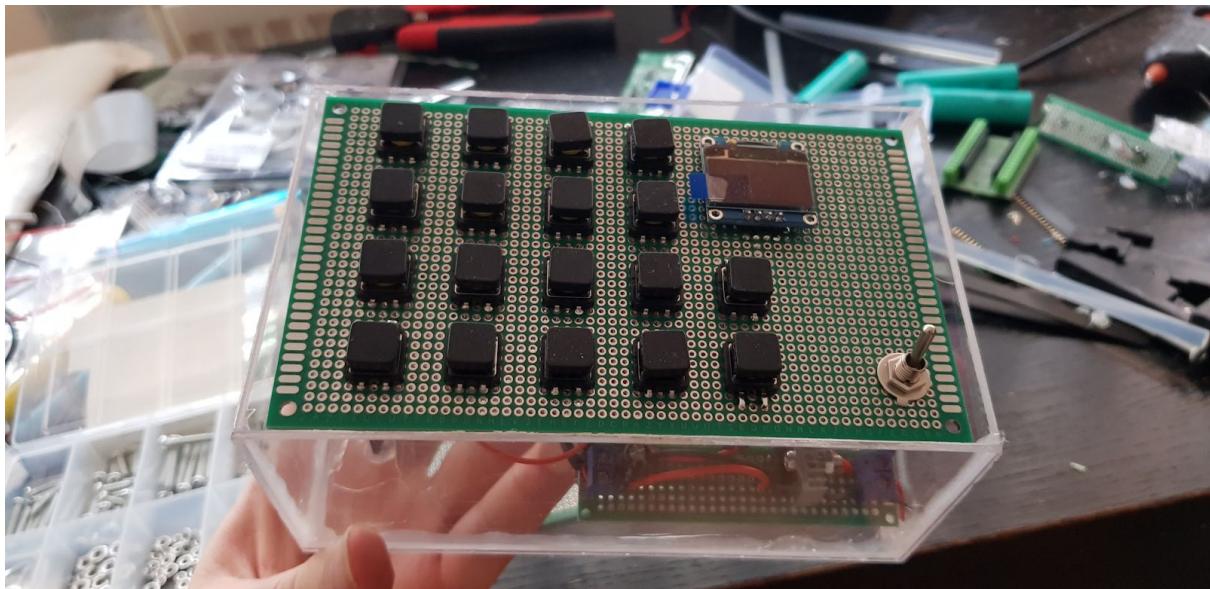
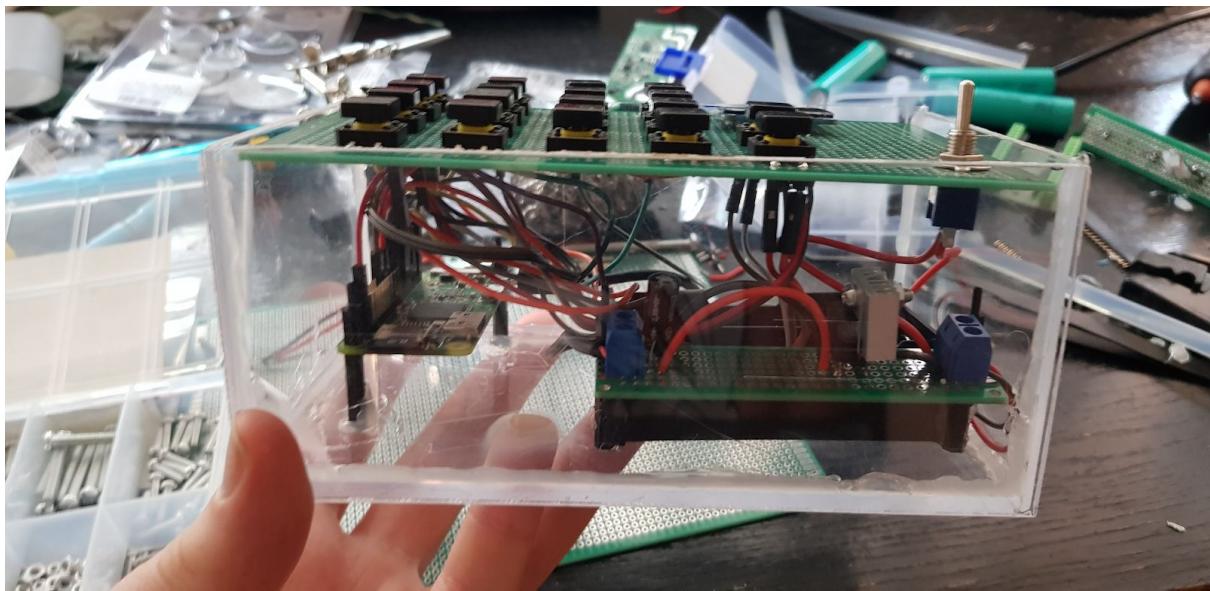
Do połączenia wszystkich elementów użyłem kleju na gorąco, jest on w zupełności wystarczający do tego stopnia, że jak ktoś się pomyli to praktycznie nie da się po ostygnięciu usunąć go w całości z plexi.



Ostatnim krokiem jest połączenie wszystkich elementów razem za pomocą kabli z końcówkami żeńska-żeńska i zamknięcie całości w obudowie. Z racji tego, że moja obudowa była większa od płytki PCB, to musiałem po bokach dodać dodatkowe wsporniki aby przyczepić płytę do obudowy.

Efekt końcowy po podłączeniu całości i zamontowaniu przełącznika w płytce PCB.





## Kod do maty

Tak jak w przypadku robota, pełny kod znajdziesz w linku poniżej.

### [Kod do maty](#)

Sama instalacja kodu na malince jest dość prosta. Najpierw klonujemy repozytorium

```
git clone https://github.com/domiipl/Projekt-Inz-Pliki
```

Następnie przechodzimy do folderu "Kod - mata" komendą

```
cd Kod\ -\ mata\
```

i wpisujemy

```
npm install
```

Czekamy aż zainstalują się biblioteki i wpisujemy kolejną komendę

```
node Mata.js
```

Gdybyś jakimś przypadkiem zmienił nazwę pliku z "Mata.js", ustaw odpowiednią nazwę w komendzie wyżej

## Opis kodu

Standardowo, musimy zacząć od dołączenia odpowiednich bibliotek oraz sterowników. Będą nam potrzebne biblioteki odpowiedzialne za komunikację sieciową, komunikację z pinami malinki oraz sterownik do ekranu oled.

```
var net = require('net')
let rpio = require('rpio')
let Oled = require('sh1106-js')
```

Potrzebne będą również pewnie zmienne. Zmienna odpowiadająca za tryb robota, gdzie "0" odpowiada trybowi "drive" a "1" trybowi "freeDrive" w kodzie robota.

```
let moveMode = 0
```

Następnie zmienna, która odpowiada za aktualnie("true") i ostatnio("false") wciśnięty przycisk, celowo jest to maksymalnie jeden przycisk.

```
let pressed = { x: 0 , y: 0, state: false }
```

Oraz zmienna, która zawiera tablicę znaków do jazdy swobodnej ("freeDrive"). Przypisujemy przycisku do znaków, jeżeli znak jest pusty, to nie zostanie wysłany do robota.

```
let freeDrive = [
  [ "", "", "", "" ],
  [ "", "", "", "" ],
  [ "", "z", "", "" ],
  [ "v", "x", "c", "" ]
]
```

Inicjalizujemy bibliotekę od Raspberry, ustawiamy opcję ekranu i je również inicjalizujemy.

```
rpio.init({
  gpiomem: false
})
var opts = {
  rpio,
  width: 128,
  height: 64,
  address: 0x3C,
  device: '/dev/i2c-1'
};
var oled = new Oled(opts)
oled.turnOnDisplay()
oled.fillRect(0, 0, 128, 64, 0)
```

Dodajemy zmienną, która odpowiada za piksele strzałek na wyświetlaczu.

```

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
],
[ // góra
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0],
[0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0],
[0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0],
[0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0],
[0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0],
[0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0],
[0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0],
[0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0],
[0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0],
[0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0],
[0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0],
[0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0],
[0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
]
,
```

Pozwolę sobie nie wklejać tu jeszcze tablic dla strzałki "w dół", "w lewo" oraz "w prawo".

Następnie funkcje, dzięki którym odpowiednio narysujemy dany symbol na wyświetlaczu.

```

function drawSymbol(x, y, arr) {
    let tab = []
    for(let i in arr) {
        for(let j in arr[i]) {
            tab.push([(parseInt(x)+parseInt(j)), (parseInt(y)+parseInt(i)), arr[i][j]])
        }
    }
    oled.drawPixel(tab)
}
function drawButtonSymbol(i, j, id) {
    let x = j * 15 + 1
    let y = i * 15 + 1
    drawSymbol(x, y, symbols[id])
}
var font = require('oled-font-5x7');
```

Teraz odpowiednio przypisujemy piny malinki. Dla rzędów i kolumn,

```

let ROWS = [7, 11, 13, 15]
let COLS = [16, 12, 10, 8]
```

oraz dla dodatkowych przycisków (Start, Zmień tryb).

```

let EROWS = [21, 23]
let ECOL = 19
```

Potrzebna będzie jeszcze tablica znaków odpowiadających za ruchy,

```
let states = [' ', 'u', 'd', 'l', 'r']
```

oraz tablica reprezentująca wartości maty z tablicy wyżej (states).

```

let values = [
    [' ', ' ', ' ', ' '],
```

```
[ ' ', ' ', ' ', ' ' ],
[ ' ', ' ', ' ', ' ' ],
[ ' ', ' ', ' ', ' ' ]
]
```

Istotnym jest również sprawdzenie, czy znaleziono robota, dlatego tworzymy zmienną "strażnik", która to sprawdza.

```
let connected = false
```

Następnie tworzymy klienta do wysyłania i odbierania socketów.

```
client = new net.Socket();
client.connect(12345, '192.168.1.200', function() {
    console.log('Connected')
    connected = true
})
```

Tworzymy tablicę, która będzie nas informować o tym czy przycisk jest wciśnięty. ("0" - nie, "1"- tak)

```
let buttons = [
    [0, 0, 0, 0],
    [0, 0, 0, 0],
    [0, 0, 0, 0],
    [0, 0, 0, 0]
]
```

Oraz identyczna tablica dla dwóch dodatkowych przycisków.

```
let buttons2 = [0, 0]
```

Następnie inicjalizujemy piny odpowiedzialne za klawiaturę 4x4,

```
for(let i in ROWS) {
    console.log("row: " + ROWS[i])
    rpio.open(ROWS[i], rpio.INPUT, rpio.PULL_DOWN)
}
for(let i in COLS) {
    rpio.open(COLS[i], rpio.OUTPUT, rpio.LOW)
}
```

oraz klawiaturę 2x1.

```
for(let i in EROWS) {
    console.log("erow: " + EROWS[i])
    rpio.open(EROWS[i], rpio.INPUT, rpio.PULL_DOWN)
}
ripi.open(ECOL, rpio.OUTPUT, rpio.LOW)
console.log("Initialised")
```

Uruchamiamy główną pętle programu.

```
checkPins()
```

Przechodzimy do głównej funkcji programu.

```
function checkPins() {
```

Najpierw dodajemy obsługę klawiatury 4x4

```
for(let i in COLS) {
    rpio.write(COLS[i], rpio.HIGH)
    for(let j in ROWS) {
        if(rpio.read(ROWS[j]) == 1) {
            if(buttons[j][i] == 0) {
                buttons[j][i] = 1
                pushButton(j, i)
            }
            else {
```

```

        pressedButton(j, i)
    }
}
else if(buttons[j][i] == 1) {
    buttons[j][i] = 0
    releaseButton(j, i)
}
}
rpio.write(COLS[i], rpio.LOW)
}

```

oraz klawiatury 2x1.

```

rpio.write(ECOL, rpio.HIGH)
for(let i in EROWS) {
    if(rpio.read(EROWS[i]) == 1) {
        if(buttons2[i] == 0) {
            buttons2[i] = 1
            pushExtraButton(i)
        }
    }
    else if(buttons2[i] == 1) {
        buttons2[i] = 0
        releaseExtraButton(i)
    }
}
rpio.write(ECOL, rpio.LOW)
//cycle++
setTimeout(function(){ checkPins() }, 1)
}

```

Potrzebna jest również funkcja, która wykonuje się po wciśnięciu przycisku przez użytkownika. Funkcja ta dostaje odpowiedni znak, w zależności który przycisk został wciśnięty i następnie rysuje odpowiedni symbol na ekranie.

```

function pushButton(i, j) {
    console.log("Button (" + i + ", " + j + ") pushed")
    if (moveMode == 0) {
        let id = states.indexOf(values[i][j])
        id++
        if(id == states.length) {
            id = 0
        }
        values[i][j] = states[id]
        drawButtonSymbol(i, j, id)
        console.log("Values after (" + i + "," + j +") press: \n" + values)
    }
}

```

Następnie potrzebujemy funkcji, która będzie się wykonywać kiedy przycisk z klawiatury 4x4 zostanie wciśnięty i przytrzymany w tym stanie. Funkcja ta służy do trybu jazdy swobodnej.

```

function pressedButton(i, j) {
    if(moveMode == 1) {
        if(pressed.x == i && pressed.y == j) {
            if(freeDrive[i][j] != "") {
                client.write(freeDrive[i][j])
            }
        }
    }
}

```

```

        else if(pressed.state == false) {
            pressed.x = i
            pressed.y = j
            pressed.state = true
        }
    }
}

```

Kolejna z funkcji, wykona się po puszczeniu trzymanego przycisku z klawiatury 4x4.

```

function releaseButton(i, j) {
    if(pressed.x == i && pressed.y == j && pressed.state) {
        pressed.state = false
    }
    console.log("Button (" + i + ", " + j + ") released")
}

```

Dodatkowo funkcja, która przywraca narysowane strzałki. **TU BĘDZIE ZMIANA**

```

function drawActualArrows() {
    for(let i in values) {
        for(let j in values[i]) {
            let id = states.indexOf(values[i][j])
            if(id != -1) {
                drawButtonSymbol(i, j, id)
            }
        }
    }
}

```

Przedostatnia z funkcji, która wykonuje się po wciśnięciu przycisku z klawiatury 2x1.

Odpowiedzialna za zmianę trybu oraz przycisku start.

```

function pushExtraButton(i) {
    console.log("Extra button(" + i + ") pushed")
    if(i == 0) {
        if(moveMode == 0) {
            moveMode = 1
            oled.fillRect(0, 0, 128, 64, 0)
            // // sets cursor to x = 1, y = 1
            oled.setCursor(10, 10);
            oled.writeString(font, 1, 'Free Drive', 1, true);
        }
        else {
            moveMode = 0
            oled.fillRect(0, 0, 128, 64, 0)
            drawActualArrows()
        }
    }
    else if(i == 1) { //start button
        let s = ''
        for(let i in values) {
            for(let j in values[i]) {
                s += values[i][j]
            }
        }
        client.write(s)
        console.log("TCP packet sent: " + s)
    }
}

```

```
}
```

Na koniec funkcja, wykonująca się po puszczeniu przycisku z klawiatury 2x1.

```
function releaseExtraButton(i) {  
    console.log("Extra button (" + i + ") released")  
}
```

## 5. Połączenie robota oraz maty

### Opis działania

W naszym projekcie zdecydowaliśmy, że to robot(Arduino) będzie tworzyć sieć wifi i ją konfigurować. Definiujemy port, server,

```
#ifndef PORT  
#define PORT 12345  
#endif  
WiFiServer server(PORT);  
const char *ssid = APSSID;
```

oraz adres, bramę i maskę sieci.

```
IPAddress ip(192,168,1,200);  
IPAddress gateway(192,168,1,254);  
IPAddress subnet(255,255,255,0);  
WiFi.softAPConfig(ip, gateway, subnet);  
WiFi.softAP(ssid);  
IPAddress myIP = WiFi.softAPIP();  
Serial.print("AP IP: ");  
Serial.println(myIP);  
server.begin();
```

Mata(Raspberry PI) jest w stanie połączyć się z usługą wpisując odpowiedni port oraz adres ip do sieci.

```
let connected = false  
client = new net.Socket();  
client.connect(12345, '192.168.1.200', function() {  
    console.log('Connected')  
    connected = true  
})
```

Kiedy wykonamy połączenie, mata będzie w stanie wyświetlić tekst(string), który zawiera zdefiniowane komendy dla robota.

```
else if(i == 1) {  
    let s = ''  
    for(let i in values) {  
        for(let j in values[i]) {  
            s += values[i][j]  
        }  
    }  
    client.write(s)  
    console.log("TCP packet sent: " + s)  
}
```

Robot odpowiednio odczyta dany ciąg komend i wykona instrukcje zgodne z nimi.

## 6. Podsumowanie

Dotarliśmy do końca naszego poradnika. Mamy nadzieję, że spodobał się wam nasz projekt i nie mieliście większych problemów podczas budowy. Poniżej zamieścimy jeszcze listę dodatkowych pomysłów, których nie udało nam się zrealizować.

### Dodatkowe pomysły

- **Obudowa robota** -
- **Dodatkowe czujniki** - naszym kolejnym pomysłem było dołożenie czujników, dzięki którym robot wykrywałby przeszkody na drodze.
- **Emulator maty** - w tym przypadku, jeśli nie masz ochoty budować maty ale interesuje Cię sam robot. Możesz napisać prostą apkę, która emuluje matę. Tym sposobem będziesz w stanie kontrolować swojego robota z aplikacji na komputerze.