ARDUINO PSA

Uruchomienie

W pierwszej kolejności należy dokonać zakupu kilku elementów, które pozwolą nam stworzyć interfejs za pomocą którego będzie możliwe wykonywanie kodowania BSI w grupie PSA, ale również będzie możliwe wykonanie kilka dodatkowych operacji, takich jak odczyt kody PIN ze sterowników silnika przy braku kluczy do auta, ale również odczyt PIN po OBD gdy posiadamy przynajmniej jeden sprawny klucz.

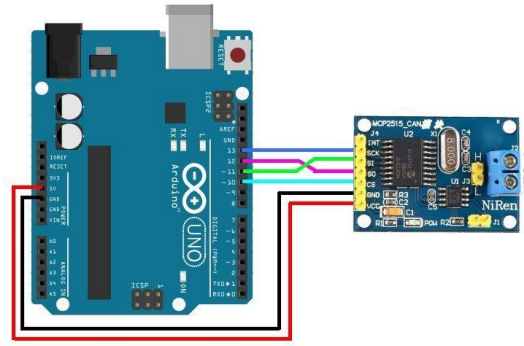
Za pomocą ARDUINO można stworzyć dwie wersje zestawu.

ARDUINO

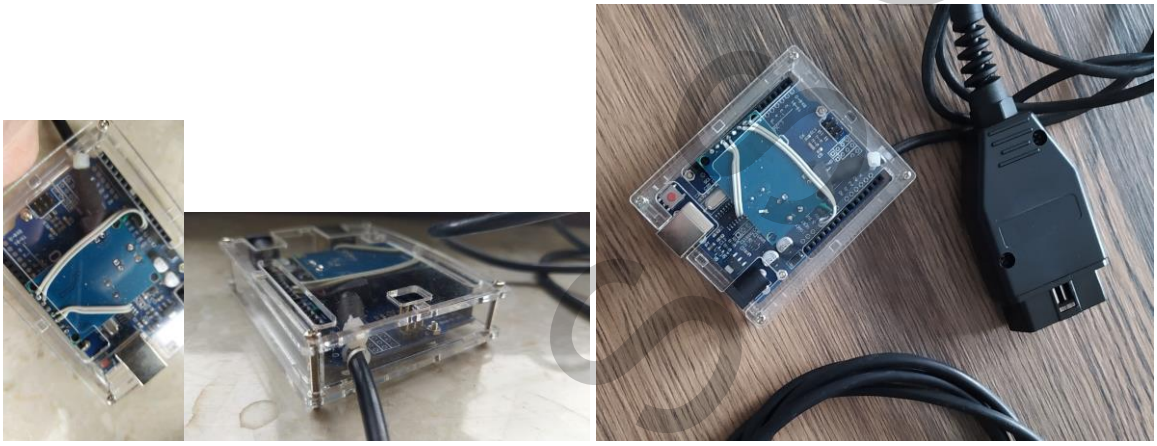
LUB

Z CAN BUS SPI MCP2515 = 8MHz

Tutaj wymagany jest dodatkowy schemat podłączenia



Po zmontowaniu



Aby mogło się to zmieścic w pojedyńczej obudowie dedykowanej do arduino jak na moim przykładzie

Należy zmodyfikować moduł CAN:

Zdemontować złacze zaciskowe przewodów (niebieskie)

Usunać piny zworek

Usunać żółty separator pinów

Usunac piny zasilania i krosowane - Zostawiamy tylko PIN (SCK , SO) – i wprowadzamy je w odpowiedniki ARDUINO

Uzyskujemy dzieki temu stabilne trzymanie płytki – a pozostałe połączenia wykonujemu kabelkami.

Oczywiście – to jest tyklko przetestowana propozycja – można to wykonać całkowicie indywidualnie – jak komu się podoba.

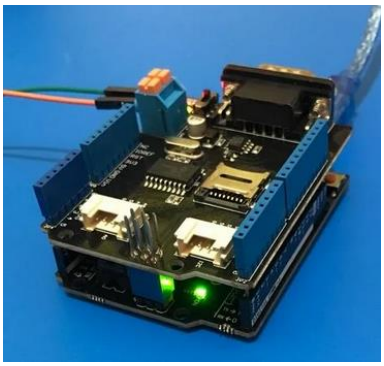
ALBO

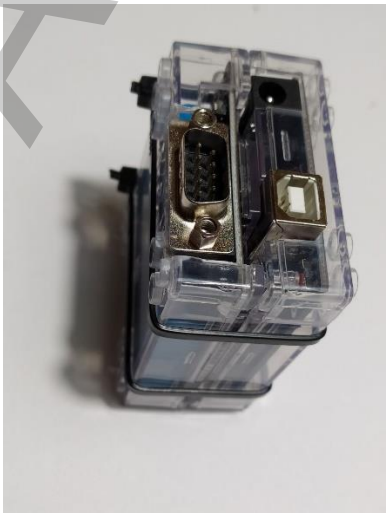
Z CAN BUS SPI MCP2515 = 16MHz

Występuje kilka modeli (z różnymi dodatkowymi funkcjami – chodzi głównie o to że rozbudowa odbywa się na zasadzie tworzenia tzw. kanapki)

Montaż nie wymaga dodatkowego opisu gdyż elementy są po prostu nakładane na siebie.

Po zmontowaniu





URUCHAMIANIE

Aby odpalić Arduino musimy do komputera wgrać program do obsługi Arduino i zainstalować sterowniki.

Pobieramy np. z tego linka:

https://www.arduino.cc/en/software (win7 i nowsze) (JUST DOWNLOAD)

Może się tak wydarzyć, że będą problemy ze sterownikiem

Przejdź wtedy do katalogu w którym zainstalowało się Arduino

Otwórz katalog Drivers

I uruchom odpowiedni sterownik.

Czasami może być wymagana aktualizacja frameworka.

Po zainstalowaniu sterowników otwieramy program Arduino i sprawdzamy czy mamy wybór Arduino na odpowiednim porcie COM – zaznaczamy ten port.

Tutaj korekta opisu – bo miałem pytania od całkowitych laików Arduino – jaki program?

Programy dla Arduino posiadają rozszerzenie .ino

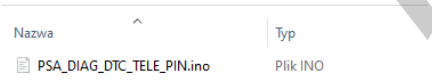
Po zainstalowaniu oprogramowania ze wskazanej strony – projekty z tym rozszerzeniem zaczną być rozpoznawane w systemie Windows. Jeżeli używamy do kompilacji Arduino.IDE – program z linku

Wsady otrzymają jedna z tych ikon

tak że po ich kliknięciu (dwukrotnym) będą się automatycznie otwierać

Ja obecnie używam Visual Studio Code (jest przyjemniejszy w pisaniu kodu – ma lepszy interfejs graficzny i pomocowy) – ale do wgrywania wsadu idealny będzie oryginalny program jaki wcześniej wskazałem.

Wsady pobieracie z linków ode mnie – maja one wskazane już rozszerzenie .ino



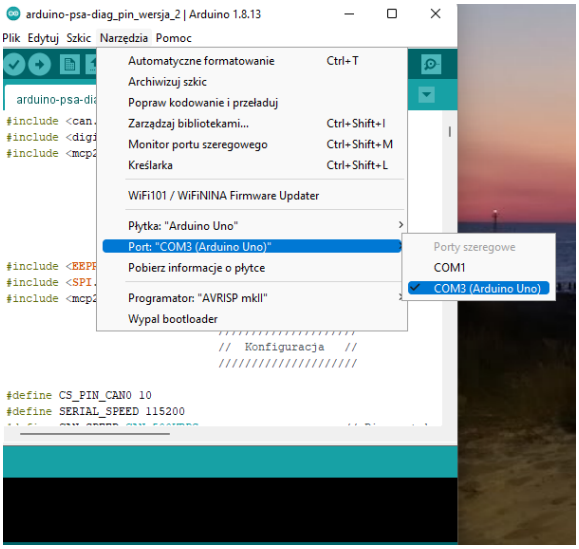
Uwaga do obrazka jest taka ze brak tu ikony Arduino.IDE – bo używam innego programu do edycji.

W programie Arduino jest tak, ze plik .ino musi znajdować się w katalogu o tej samej nazwie i najlepiej pobrane spakowane pliki wypakować i nie zmieniać nazw katalogu.

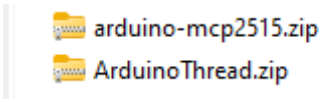
Jeżeli wyjmiecie plik .ino z katalogu po jego uruchomieniu otrzymacie monit o takiej potrzebie umieszczenia pliku w katalogu o tej samej nazwie – po zatwierdzeniu program wykona te operacje automatycznie.

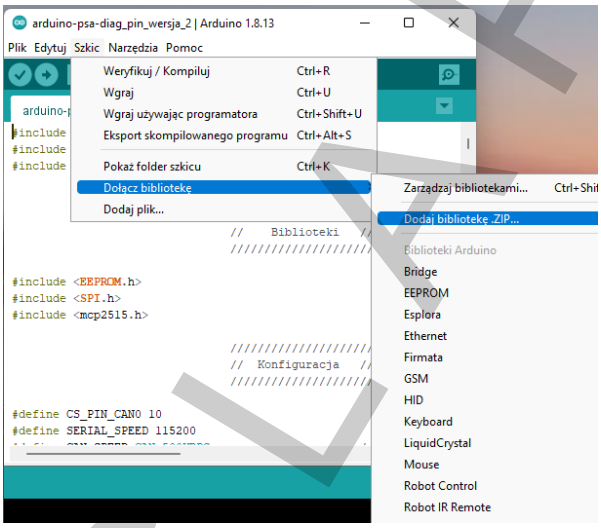
Następnie podpinamy Arduino do USB i wskazujemy w programie port COM – ktoś zapyta dlaczego ?

Przecież program wykrywa Arduino – no tak ale jak ktoś pracuje z kilkoma płytkami to musi wskazać na która przesłać wsad.



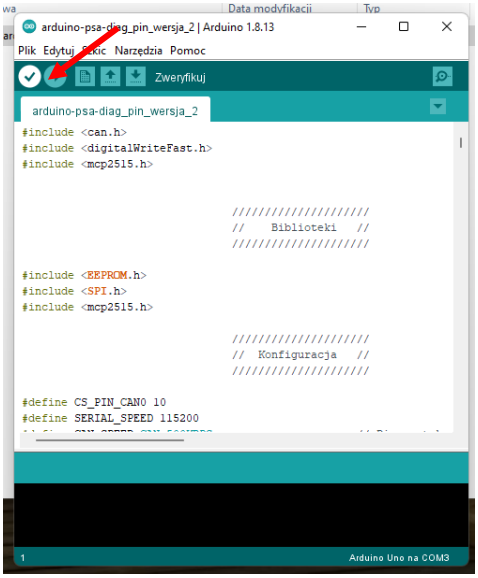
Następnie wgrywamy biblioteki do obsługi płytki CAN – Wybieramy zakładkę szkic i zależnie od posiadanych plików (najczęściej jest to zip) wgrywamy bibnlioteki.



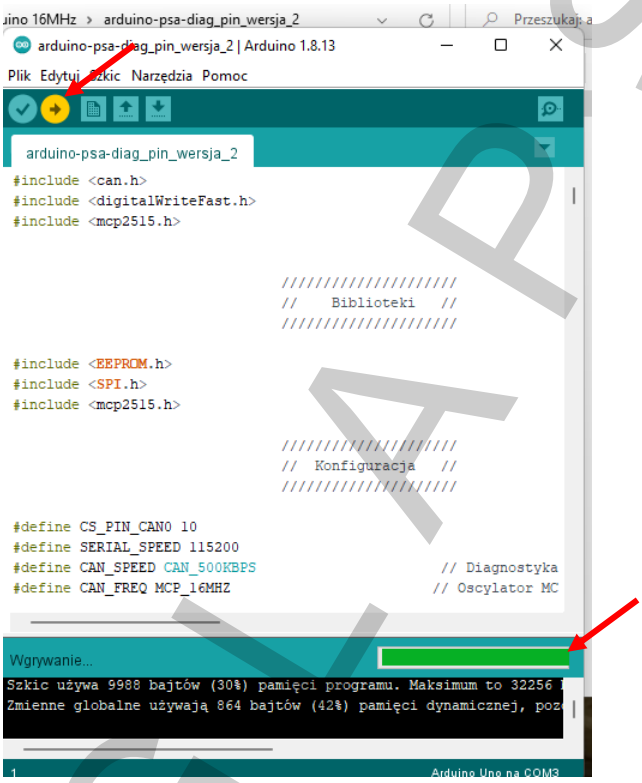


Jeżeli używasz plików z moich linków to zawierają one odpowiednie biblioteki w katalogach do wybrania.

Po dołączeniu bibliotek weryfikujemy zawartość

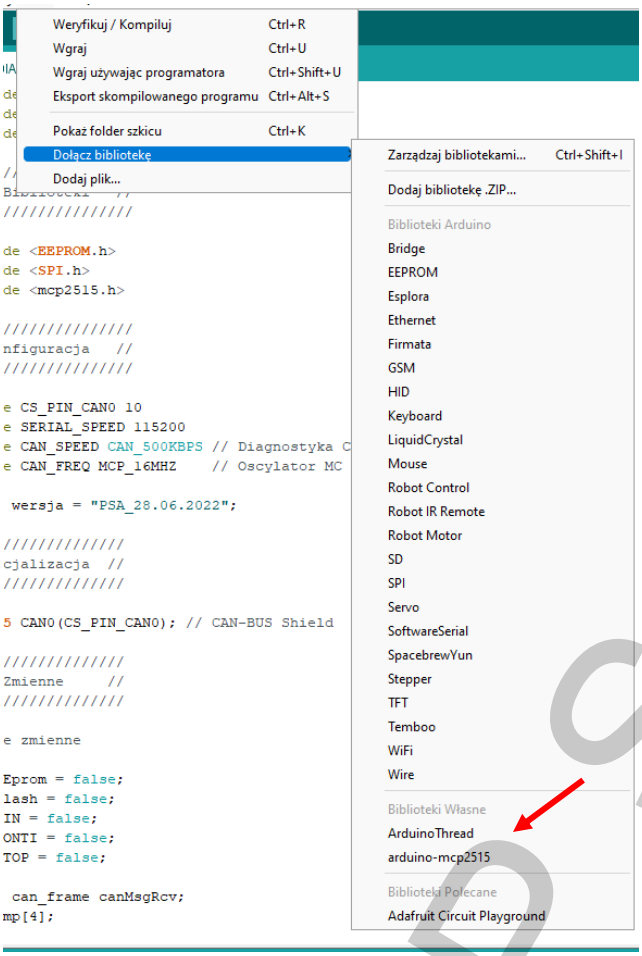


Jak wszystko przechodzi OK to dokonujemy wgrania programu do arduino



I tutaj uwaga do Bibliotek

Po ich dołączeniu widoczne one są na tym drop down-ie



W kolejnych poprawkach o ile takie naniosę – dla wsadu Arduino podczas ich wgrywania – nie ma konieczności dołączania bibliotek o ile one już wcześniej zostały dołączone i są widoczne w tym pasku.

Jeżeli jednak to zrobicie to program poinformuje ze takowe już są dołączone.

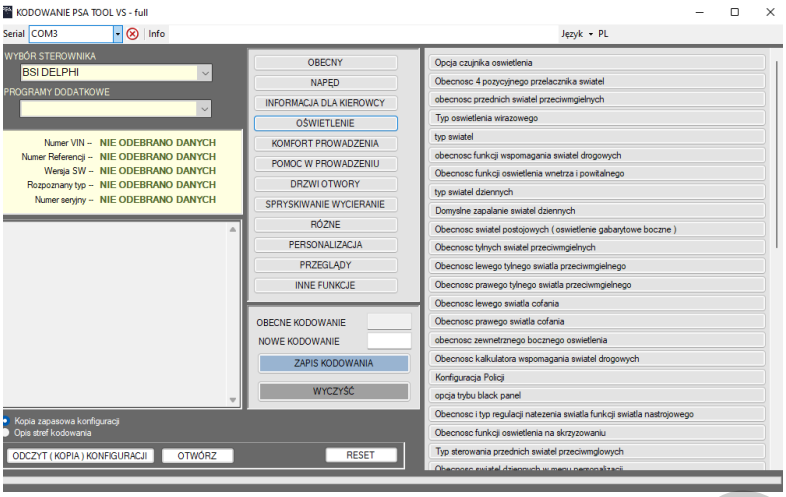
W plikach ode mnie występują dwa wsady Arduino zależnie od użytej płytki CAN

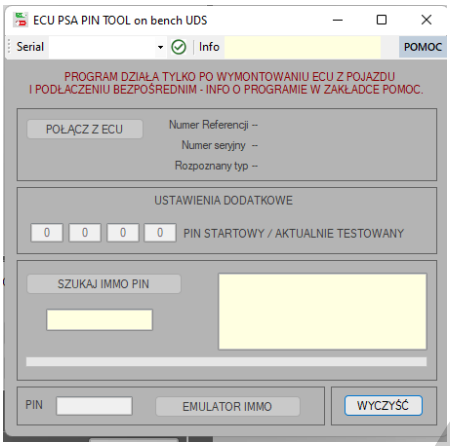
8MHz lub 16MHz

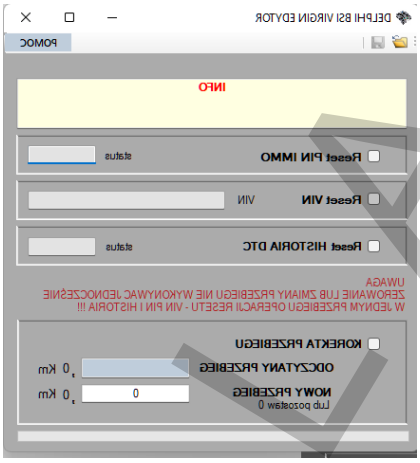
Należy wgrać zgodne oprogramowanie aby CAN działał prawidłowo.

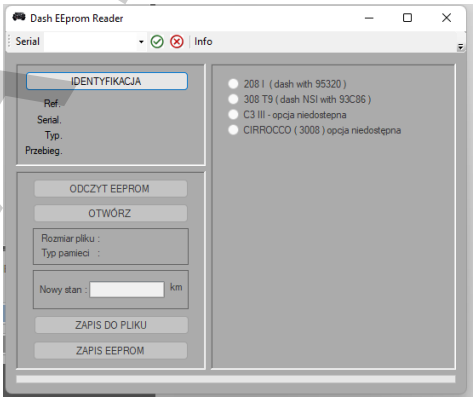
Po poprawnym uruchomieniu Arduino można cieszyć się już działaniem wszystkich programów, które obecnie oferuję i które może w przyszłości się pojawią.

Program ewoluuje i poniższe screeny mogą różnic się od aktualnie oferowanego zestawu





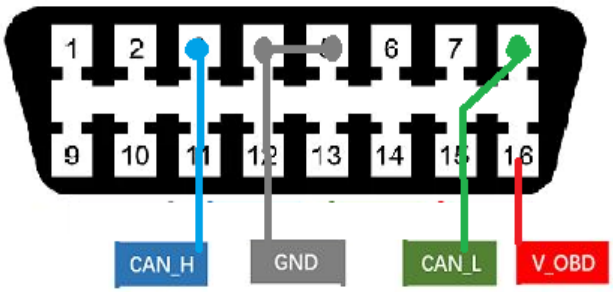




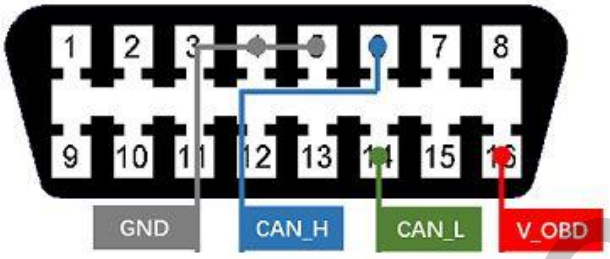
Musze również dodać kilka słów o złączu OBD

W tańszym zestawie jeżeli podążaliście moją droga to w miejsce wylutowanego niebieskiego zacisku CAN należy dolutować kabelki CAN zgonie z opisem.

W kwestii zaś samego złącza OBD to diagnostyka w grupie PSA odbywa się na pinach 3 i 8.

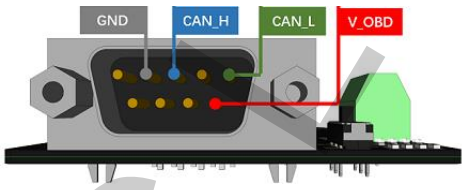


Wyjątkiem będzie sytuacja kiedy będziemy chcieli używać mojego oprogramowania do podsłuchiwania CAN sterownika silnika w celu odczytu PIN gdy posiadamy przynajmniej jeden kluczyk z auta – wtedy należy CAN doprowadzić na piny 6 i 14



Jeżeli chcemy jednej wtyczki używać do obydwu podłączeń to warto by było pomyśleć o przełączniku we wtyczce OBD to przełączania linii CAN – ale to kwestia indywidualnego podejścia.

Jeżeli chodzi o płytkę CAN Arduino z wyprowadzeniem DB9 – pinout przedstawia się następująco



MULTIPLEXER

Istnieje możliwość rozbudowy interfejsu o dodatkowy układ który będzie przełączał piny 6,14 na 3,8

W programie obecnie jest już zaimplementowana taka możliwość.

Użyte zostały piny ARDUINO 5 i 6

Sterowanie przechodzi z programu PSA TOOL po wybraniu odpowiedniej funkcji

Na pinach pojawia się stan wysoki lub niski .

Przy opcji PIN z auta przy 1 kluczu na pinach 5 i 6 arduino pojawia się sygnał wysoki

Co umożliwia wysterowanie płytki z przekaźnikami

Np. takiej:

Złącza DC+ i DC- zasilamy z Arduino 5V

IN1 – pin 5 Arduino

IN2 – pin 6 Arduino

Przełaczanie CAN

Do środkowych pinów wyprowadzeń podłączamy CAN L i CAN H

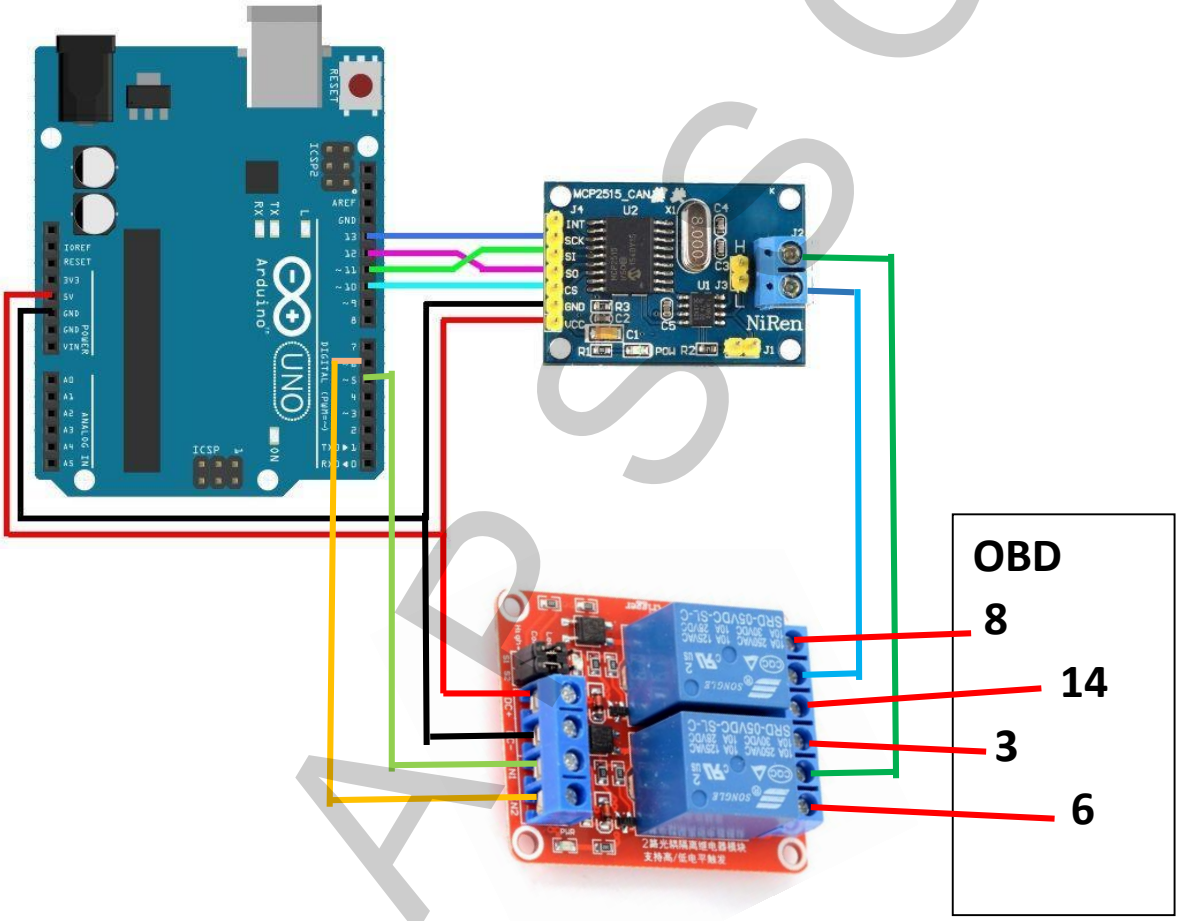
I zależnie od stanu możemy wyprowadzić to na różne PINY OBD

Przykładowe podłączenie z użytymi elementami:

Oczywiście dla łatwiejszego rysowania wyprowadzenia OBD wyglądają tak a nie inaczej

Ale sprawa jest dowolna

Oby tylko CAN L trafiał w CAN L a CAN H w CAN H



W takim układzie nie będzie konieczne stosowania przełącznika we wtyczce OBD

Jest to rozwiązanie droższe i wymagające większej obudowy.

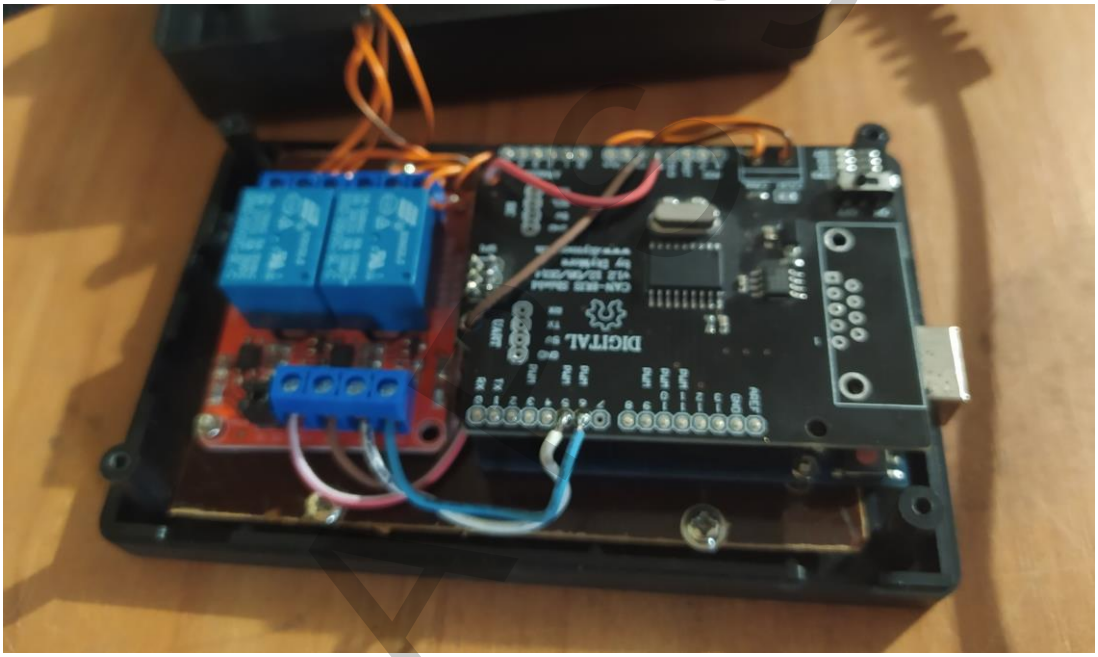
Może niezbyt profesjonalny – bo nie istnieje odpowiednia obudowa – ale po zdjęciu już zbędnych rzeczy z płytki CAN

Wszystko zmieściło się z luzem w taka oto obudowę: KM-40 130x80x35mm ABS
Usunięto dla tej obudowy i płytki CAN shield 2.0 16MHz :

Złącze DB9
Złącza i piny kontaktowe
Złącze zaciskowe (CAN)
Obcięto nosek przełącznika

Z mała płytka CAN 8MHz – te operacje nie będą konieczne.

Zamontowano na dodatkowej płytce gdyż otwory nie pasują jakby się chciało.
Niestety na miejscu nie udało się dostać czarnego dławika fi7 zatem biały.
Za to funkcjonalność 100% - pełna ochrona zawartości – gdyż obudowy Arduino nie mogą pochwalić się taka trwałością – a do tego Automatyczne działanie na pinach OBD.



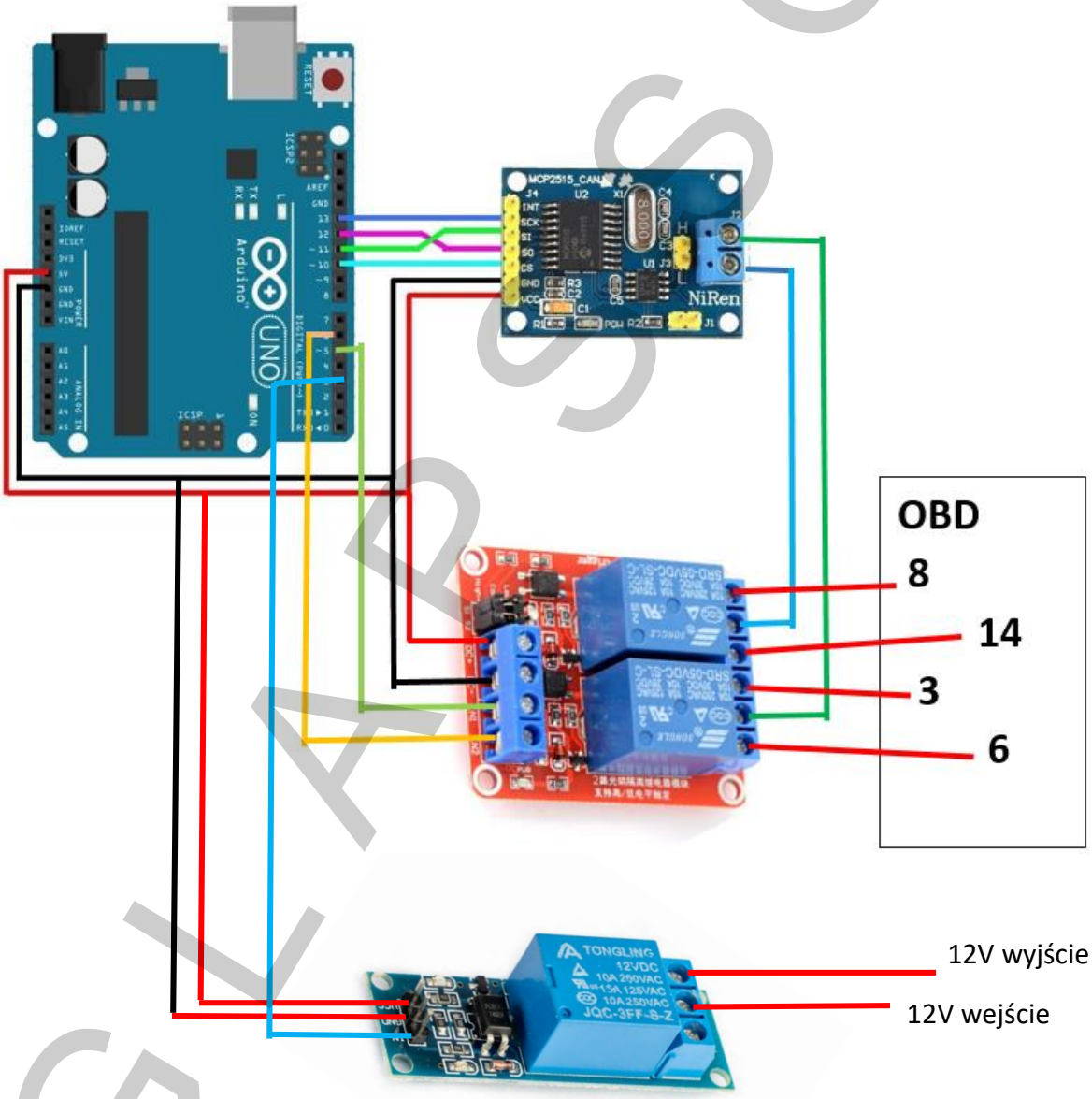
W niektórych aplikacjach lub przy pracy z niektórymi sterownikami wymagany jest restart zasilania w obecnej wersji przewidziana jest taka możliwość z wykorzystaniem ARDUINO

Wyprowadzenie dla przekaźnika restartu zostało przygotowane jak na poniższym schemacie

Obecnie program jest tak zorganizowany ze stan wysoki jest podawany tylko wtedy gdy zostanie to wywołane z programu – normalnie przekaźnik jest w stanie spoczynkowym.

Dalsza rozbudowę można poszerzyć do takiego układu:

Można od razu zastosować układ przekaźników wielokanałowy np. 3 kanały zamiast jak poniżej 2 +1



OPCJONALNIE-ZASTOSOWANIE PRZEKAŹNIKA RESET DLA FUNKCJI PROGRAMU

Możliwość użycia została ujęta od wersji oprogramowania `220810_MPx`