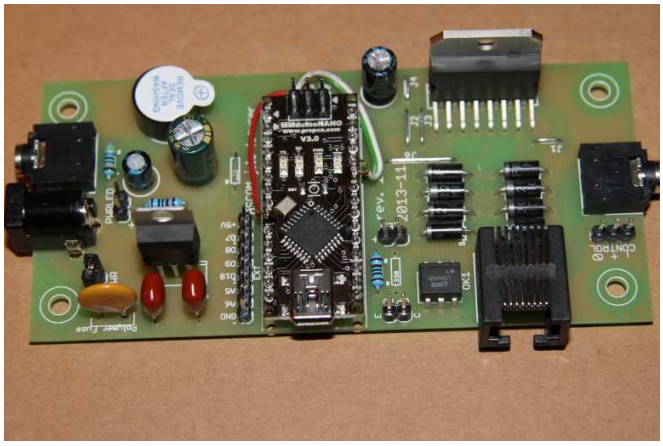
ASCOM Jolo Focuser wersja 1.5



UWAGA

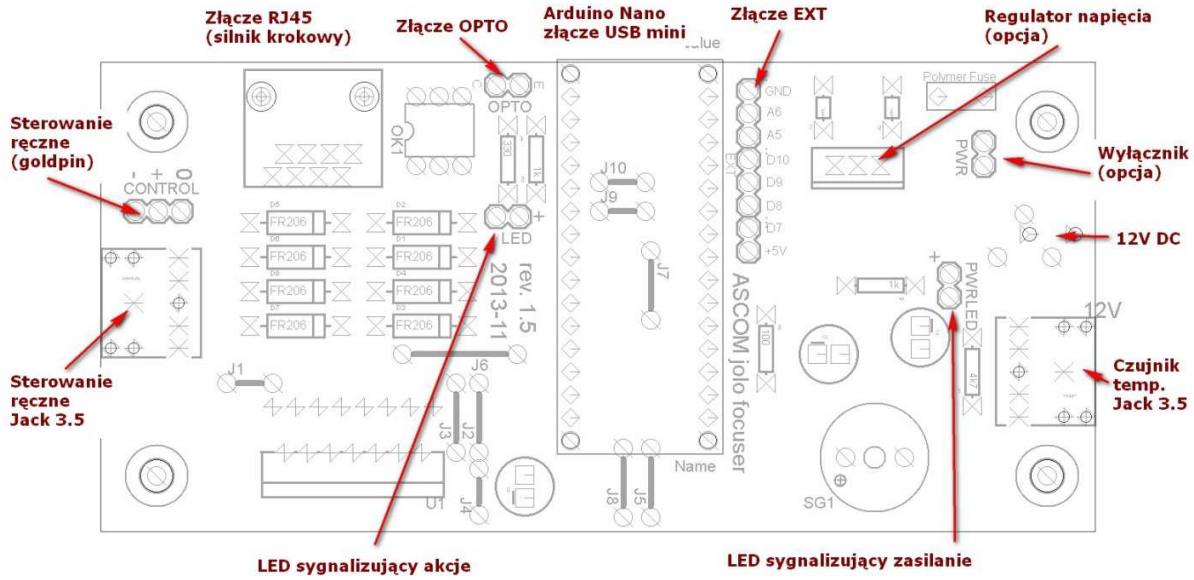
Gniazdo zasilające urządzenie w wersji 1.5 posiada minus w środku.

Opis gniazda EXT zawiera pomyłkę zamiast A5 powinno być A7

Opis gniazda OPTO zawiera pomyłkę – zamieniono miejscami piny E i C

Opis układu

Układ oparty jest na platformie Arduino Nano 3.0 (http://arduino.cc/en/Main/ArduinoBoardNano) która odpowiada za sterowanie poszczególnymi komponentami urządzenia oraz za komunikację z komputerem przy użyciu wbudowanego interfejsu USB-RS232 FTDI. Po stronie komputera zainstalowany musi być sterownik ASCOM (do pobrania ze https://github.com/sirJolo/ascom-jolo-focuser/tree/master/Downloads) który pośredniczy w komunikacji pomiędzy układem focusera oraz aplikacjami sterującymi ustawianiem ostrości (*Maxim DL, Focus Max, APT* i inne umożliwiające sterowanie urządzeniami opartymi o interfejs ASCOM).



Rysunek przedstawia płytkę urządzenia z zaznaczonymi gniazdami i złączami do wykorzystania przez użytkownika.

1. Gniazdo zasilania 5.5/2.1mm. Urządzenie może być zasilane napięciem z zakresu od 7 do 16V
2. Gniazdo jack 3.5mm stereo do podłączenia czujnika temperatury DS1820
3. Gniazdo USB mini (umieszczone na płytce Arduino Nano) zapewniające komunikację z komputerem
4. Gniazdo RJ45 posiadające 8 wyprowadzeń połączonych w cztery pary po dwa wyprowadzenia sterujące bipolarnym silnikiem krokowym
5. Gniazdo jack 3.5mm stereo do podłączenia dwóch przycisków do sterowania ręcznego.

Oprócz wyżej wymienionych gniazd na płytce znajdują się dodatkowo złącza:

6. Złącze PWR do podłączenia dodatkowego wyłącznika zasilania
7. Złącze PWRLED do podłączenia diody LED sygnalizującej zasilanie
8. Złącze 8 pinowe EXT podłączone do wyprowadzeń modułu Arduino do wykorzystania przez użytkownika
9. Złącze OPTO izolowane optycznie do sterowania dowolnym urządzeniem (na przykład migawką aparatu fotograficznego)
10. Złącze do podłączenia diody LED sygnalizującej pracę urządzenia
11. Złącze do podłączenia przycisków do sterowania ręcznego, które można umieścić np. wewnątrz obudowy urządzenia

Po podłączeniu urządzenia do portu USB komputera powinno ono zostać rozpoznane i po zainstalowaniu sterownika ASCOM będzie ono widoczne w aplikacjach przy wybieraniu dostępnych urządzeń do ustawiania ostrości jako *Jolo ASCOM focuser*. W tym momencie będzie już działała komunikacja z urządzeniem, które będzie zasilane z portu USB komputera. Po podpięciu napięcia zasilania do urządzenia oraz silnika otrzymamy w pełni działający układ. Wszystkie pozostałe komponenty są opcjonalne.

Podłączenie urządzenia

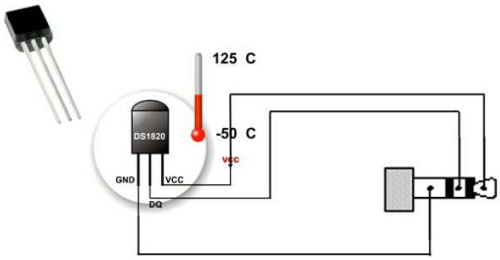
W sekcjach poniżej opisano szczegółowo sposób podłączenia poszczególnych elementów do urządzenia. Przy pierwszym uruchomieniu możemy skorzystać z poniższego scenariusza:

1. Podłączamy urządzenie kablem USB do komputera. Po chwili system wykryje urządzenie i zainstaluje odpowiedni sterownik interfejsu USB-RS232 FTDI. Jeżeli system nie zainstaluje sterowników automatycznie znajdziemy je na stronie FTDI albo w folderze *Moje Dokumenty/Arduino/drivers* o ile mamy zainstalowane Arduino IDE. Po zainstalowaniu w naszym systemie pojawi się dodatkowy port szeregowy COM
2. W urządzeniu domyślnie zainstalowane jest oprogramowanie w wersji 1.5. Jeśli chcemy je zaktualizować albo zmienić niektóre parametry możemy to zrobić w tym momencie używając środowiska Arduino IDE

3. Następnie możemy zainstalować sterownik ASCOM. Po jego zainstalowaniu z dowolnego programu obsługującego focusery ASCOM (np. *Maxim DL* albo *APT*) wybieramy *Jolo ASCOM focuser* i po otwarciu okienka ustawień wprowadzamy odpowiednie wartości
    a. Ustawiamy odpowiedni port COM
    b. Ustawiamy obliczony przesuw wyciągu odpowiadający jednemu krokowi silnika (możemy to zrobić później)
    c. Wprowadzamy maksymalną wartość kroku odpowiadającą całkowitemu wysunięciu wyciągu
4. Po wprowadzeniu tych wartości możemy już połączyć się z urządzeniem i sprawdzić reakcje na wprowadzane z oprogramowania polecenia
5. Następnie podłączamy do urządzenia zewnętrzne zasilanie i silnik. Po tej czynności polecenia wydawane z programu powinny skutkować odpowiednimi ruchami silnika krokowego
6. Kolejnym krokiem (opcjonalnym) będzie podłączenie czujnika temperatury. Po wykonaniu tej czynności program powinien wyświetlać aktualną temperaturę czujnika, który możemy umieścić w odpowiednim miejscu (na przykład przymocować do tubusu naszego teleskopu)
7. Następnym opcjonalnym krokiem może być podłączenie przycisków do ręcznego sterowania wyciągu

Czujnik temperatury

Oparty jest o popularny układ DS1820 i podłączany do gniazda jack 3.5mm na górze płytki obok gniazda zasilania. Aby układ wykrył czujnik musi on być podpięty przed włączeniem zasilania – podłączenie czujnika do włączonego urządzenia nie umożliwi nam pomiaru temperatury.
Schemat podłączenia czujnika temperatury:

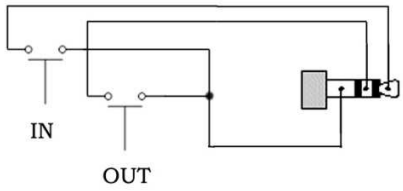


Buzzer oraz sygnalizacja świetlna

Na płytce umieszczono piezoelektryczny brzęczyk oraz złącze do podłączenia dodatkowej diody LED sygnalizującej działanie układu. Sygnalizowane są akcje takie jak: podłączenie urządzenia do komputera i nawiązanie komunikacji, zakończenie ruchu silnika, błędna komenda. W celu obniżenia głośności brzęczyka zamiast zworki R2 można zastosować rezystor o wartości 30-200Ω.

Sterowanie ręczne

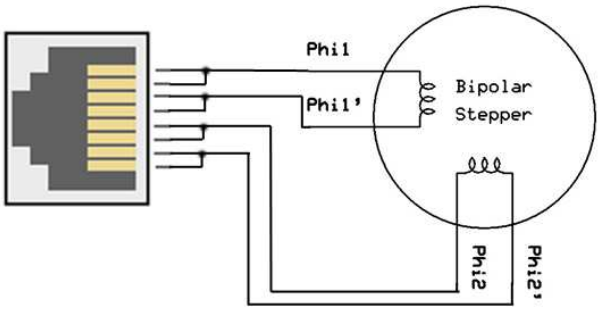
Urządzenie umożliwia podpięcie dwóch przycisków do sterowania ręcznego. Naciśnięcie przycisku powoduje uruchomienie silnika krokowego z minimalną szybkością która rośnie aż do osiągnięcia maksymalnej zadanej w konfiguracji wartości (w ciągu około 2 sekund). Po zwolnieniu przycisku silnik zatrzymuje się. Krótkie naciśnięcia przycisku umożliwiają sterowanie z dokładnością do pojedynczych kroków. Przyciski można podpiąć do gniazda jack 3.5mm na dole płytki (przyciski zewnętrzne) oraz do złącza CONTROL na płytce położonego obok tego gniazda (np. dla umieszczenia przycisków wewnątrz obudowy). Można używać obu złączy jednocześnie – przyciski będą pracowały niezależnie. Bieżące położenie focusera jest aktualizowane w urządzeniu i wyświetlane w aplikacji sterującej (o ile jest podłączona). Po osiągnięciu pozycji 0 albo maksymalnej zadanej w konfiguracji sterownika ASCOM silnik zatrzymuje się.
Schemat podłączenia przycisków sterowania ręcznego:



Silnik krokowy

Do podłączenia silnika zamontowane zostało gniazdo RJ45 z ośmioma stykami. Każde dwa sąsiadujące styki są połączone ze sobą tworząc cztery niezależne linie sterujące silnikiem bipolarnym. W urządzeniu może zostać użyty dowolny silnik krokowy bipolarny z czterema wyprowadzeniami. Silnik będzie zasilany napięciem podanym do zasilania urządzenia pomniejszonym o około 2V (taki jest spadek napięcia na układzie sterownika silnika L298). Jest możliwość użycia silnika na niższe napięcie, ale wymaga to zamontowania do płytki dodatkowego regulatora napięcia w miejscu zworki oznaczonej na pierwszym rysunku tekstem *„Regulator napięcia”*. Zasilając np. urządzenie napięciem 12V i podłączając silnik na napięcie 7V należy zastosować regulator 7808 o napięciu wyjściowym 8V, które po spadku o 2V da nam 6V na silniku. Regulator należy wyposażyć w odpowiedni radiator w zależności od prądu pobieranego przez silnik. Zasilanie silnika niższym napięciem spowoduje pewien spadek momentu obrotowego oraz maksymalnej prędkości obrotowej, ale przy różnicy 20-30% nie będzie to znaczny spadek.
Układ bez modyfikacji umożliwia sterowanie silników z maksymalnym prądem 1A. Po wymianie bezpiecznika polimerowego (obok gniazda zasilania) na większy i dołożeniu niewielkiego radiatora na układ L298 można sterować silnikami z prądem do 2A. Większe obciążenie może spowodować nadmierne nagrzewanie się ścieżek na płytce i doprowadzić do jej uszkodzenia. Oczywiście zastosowany

zasilacz musi zapewnić odpowiedni prąd.
Schemat podłączenia silnika bipolarnego do urządzenia:



Jeśli po zmontowaniu układu wyciąg porusza się w odwrotnym kierunku (a więc przy sterowaniu w kierunku *In* albo przyciskiem '-' porusza się na zewnątrz) należy zamienić końcówkami jedną z par połączeń silnika (np. *Phi1* i *Phi1'*) albo w kodzie Arduino (plik *focuser.ino*) zmienić kolejność w deklaracji silnika z

```
AccelStepper stepper = AccelStepper(AccelStepper::HALF4WIRE, A3, A4, A1, A5);
```

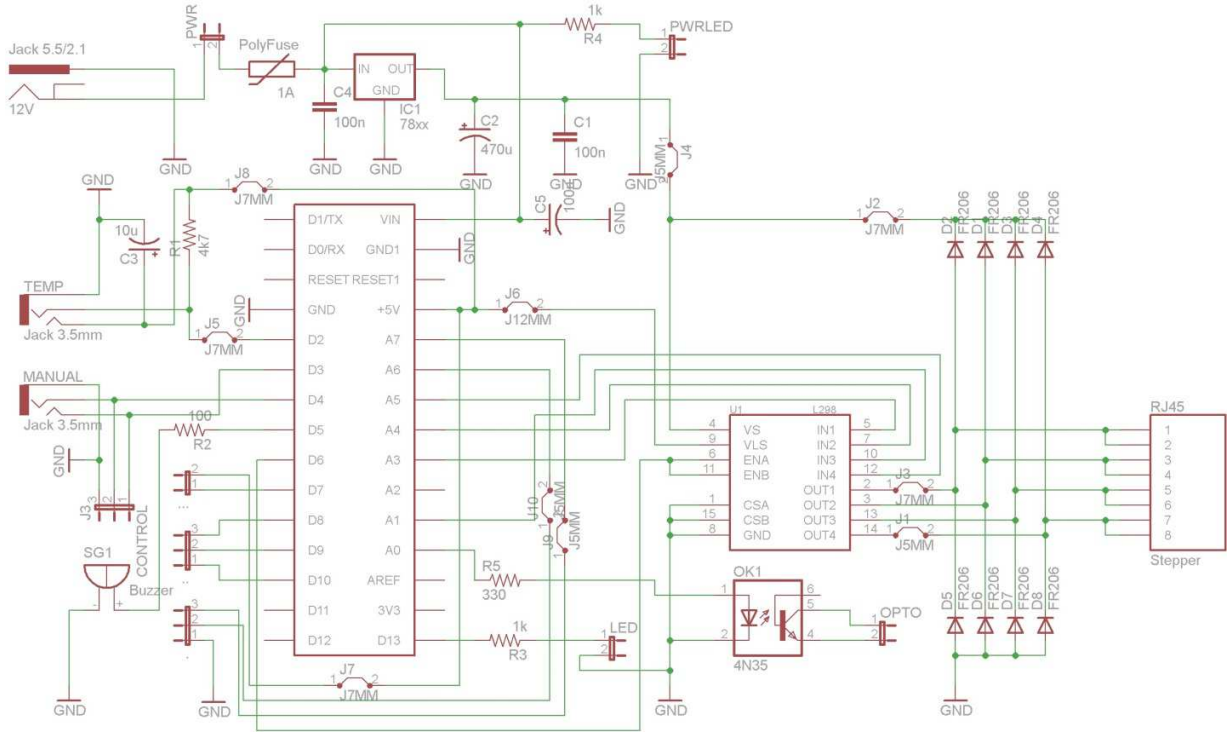
np. na:

```
AccelStepper stepper = AccelStepper(AccelStepper::HALF4WIRE, A4, A3, A1, A5);
```

Przełożenie silnika i krok focusera

Urządzenie steruje silnikiem w tzw. trybie półkrokowym. Czyli podłączając silnik o kroku równym 1.8 stopnia (200 kroków na obrót) jeden krok zostanie podzielony na dwa dodatkowe półkroki i wynikowo nasz silnik będzie wymagał 400 impulsów na pełen obrót. Dość istotnym parametrem dla automatyzacji ustawiania ostrości jest przełożenie kroku silnika na dystans pokonywany przez wyciąg. Jeśli mamy zmontowane urządzenie możemy go prosto zmierzyć suwmiarką w jednej pozycji, następnie wykonać przesunięcie na przykład o 1000 kroków i dokonać drugiego pomiaru a następnie podzielić otrzymane przesunięcie przez liczbę kroków. Możemy go również obliczyć znając całkowite przełożenie zastosowanej przekładni oraz średnicę osi przesuwającej tubus wyciągu. Zalecane jest aby rozmiar kroku był przynajmniej 3-4 razy mniejszy niż tak zwany CFZ (ang. *Critical Focus Zone*). CFZ możemy obliczyć przy użyciu np. prostego kalkulatora online pod adresem http://www.wilmslowastro.com/software/formulae.htm#CFZ . Jeśli w naszym zestawie przykładowe CFZ wynosi 80µm to pojedynczy krok silnika nie powinien przesuwać wyciągu o więcej niż 25-30µm.

Schemat ideowy focusera



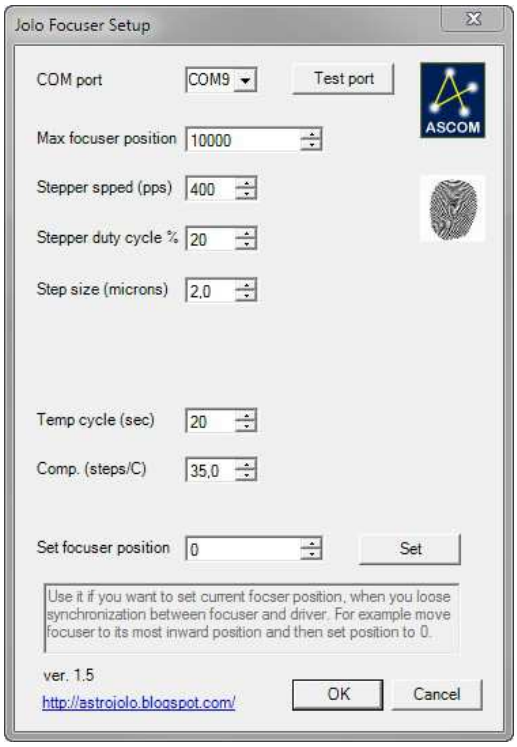
Złącze dodatkowe EXT oraz OPTO

Złącze EXT umieszczone jest obok płytki Arduino i zawiera 8 pinów. Są to patrząc od lewej masa, dwa piny analogowe, 4 piny cyfrowe oraz +5V. Umożliwiają one podpięcie do urządzenia dodatkowych urządzeń peryferyjnych takich jak np. przekaźniki włączające określone urządzenia, dodatkowe czujniki temperatury albo wilgotności, sterowanie opaskami grzewczymi na zestawie, oraz wiele innych. Napięcie +5V oraz piny złącza EXT zasilane są bezpośrednio ze stabilizatora w module Arduino. Z tego powodu nie powinno się ich obciążać sumarycznie większym prądem niż 60mA, a pojedynczego pina większym prądem niż 30mA. Dlatego sterowanie przekaźnikami czy opaskami grzewczymi musi być zrealizowane za pomocą dodatkowego klucza na tranzystorze albo na przykład układzie ULN2003 (który zawiera 7 takich kluczy).
Poniżej płytki Arduino znajduje się dwupinowe złącze oznaczone OPTO. Jest to dodatkowe złącze do sterowania na przykład migawką aparatu fotograficznego albo innego urządzenia które chcemy galwanicznie oddzielić od układu focusera. Sterowanie dodatkowymi złączami na płytce będzie wprowadzone w wersji 2.0 sterownika ASCOM, ale użytkownik może zastosować własne rozwiązania, ponieważ zarówno skecz Arduino jak i sterownik ASCOM to oprogramowanie otwarte i jest dostępne pod adresem https://github.com/sirJolo/ascom-jolo-focuser

Aktualizowanie sterownika ASCOM

Najnowsze i poprzednie wersje sterownika ASCOM do focusera można znaleźć na stronie https://github.com/sirJolo/ascom-jolo-focuser/tree/master/Downloads. W celu instalacji albo aktualizacji drivera odpowiedni instalator należy pobrać oraz uruchomić na swoim komputerze. Po zainstalowaniu należy sprawdzić czy w okienku dialogowym sterownika pojawiła się nowa wersja (w lewym dolnym rogu okienka). Jeśli wciąż jest tam poprzednia, należy odinstalować sterownik z panelu sterowania i zainstalować nową wersję „na czysto".

Opcje sterownika ASCOM



COM port – wybór portu szeregowego do komunikacji z urządzeniem

Test port – przycisk służący do sprawdzenia wybranego portu

Max focuser position – ilość kroków urządzenia odpowiadająca maksymalnemu wysunięciu wyciągu

Stepper spped (pps) – prędkość obrotowa silnika krokowego w impulsach na sekundę. Przy wartości 400 i ilości kroków silnika 200 będzie on wykonywał jeden obrót na sekundę, ponieważ jest sterowany półkrokowo

Stepper duty cycle % - moment trzymający silnik w czasie spoczynku. Przy wartości 0% silnik w czasie spoczynku nie będzie zasilany. Jeśli chcemy wykorzystać silnik do hamowania wyciągu w czasie spoczynku, dobieramy tutaj odpowiednią wartość.

Step size (microns) – przesuw wyciągu odpowiadający jednemu krokowi silnika.

Temp cycle (sec) – czas pomiędzy kolejnymi odczytami temperatury z czujnika dla kompensacji

Comp. (steps/C) – wartość kompensacji temperaturowej urządzenia. Wprowadzając np. wartość 35 dla przy zmianie temperatury o 1 stopień wyciąg przesunie się o 35 kroków. Oczywiście wartości ujemne są również dopuszczalne. Ustawienie wartości na 0 spowoduje wyłączenie kompensacji

Set focuser position – możemy ręcznie ustawić w pamięci urządzenia nową wartość pozycji. Ma to głównie zastosowanie po rozsynchronizowaniu się rzeczywistej pozycji wyciąg z tą zapamiętaną w urządzeniu, albo przy pierwszym uruchomieniu. Po przesunięciu wyciągu do pozycji minimalnej możemy ustawić w tym polu wartość 0 i po naciśnięciu przycisku *Set* będzie ona zapamiętana w pamięci wewnętrznej kontrolera

Aktualizowanie wsadu Arduino

Aby zaktualizować firmware kontrolera Arduino będziemy potrzebowali:

1. Plik *Arduino sketch x.x.zip* w odpowiedniej wersji
2. Plik *Arduino libs x.x.zip* w odpowiedniej wersji
3. Arduino IDE

Dwa pierwsze pobierzemy ze strony https://github.com/sirJolo/ascom-jolo-focuser/tree/master/Downloads , natomiast IDE pobierzemy ze strony http://arduino.cc/en/Main/Software . Po zainstalowaniu IDE biblioteki (plik *Arduino libs*) należy rozpakować do folderu *Moje Dokumenty\Arduino\libraries* i następnie uruchamiamy IDE, otwieramy plik *focuser.ino* z rozpakowanego archiwum *Arduino sketch* , kompilujemy i po podłączeniu urządzenia kablem USB do komputera ładujemy firmware do układu Arduino. Do tej operacji nie jest wymagane podłączanie zewnętrznego zasilania – moduł Arduino będzie zasilany z gniazda USB. Jeśli dla własnych potrzeb zmienialiśmy parametry konfiguracyjne wsadu Arduino, należy przed załadowaniem zmienić je również w nowej wersji (patrz sekcja poniżej).

Spis parametrów konfiguracyjnych we wsadzie do Arduino

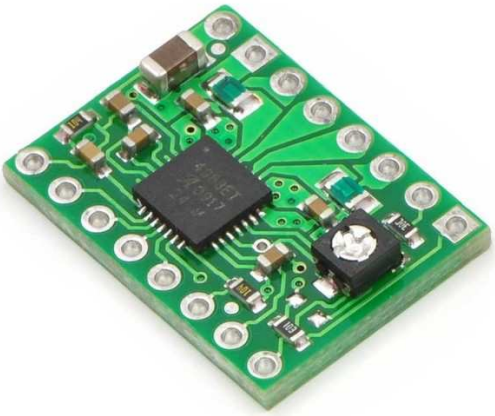
W kodzie kontrolera możemy konfigurować następujące parametry:

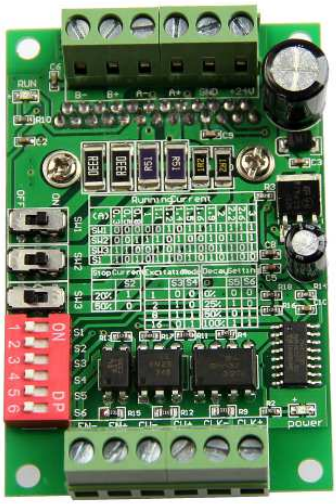
BUZZ_LONG – czas długiego sygnału brzęczyka w ms
BUZZ_SHORT – czas krótkiego sygnału brzęczyka w ms
BUZZ_PAUSE – przerwa pomiędzy sygnałami brzęczyka w ms
BUZZER_ON – włączenie/wyłączenie brzęczyka (dioda sygnalizacyjna będzie działała)
TEMP_CYCLE – częstotliwość pomiaru temperatury przez czujnik w ms
STEPPER_ACC – przyspieszenie silnika krokowego w czasie pracy kontrolowanej przez komputer w pps/s
MANUAL_STEPPER_ACC – jak wyżej ale w czasie sterowania ręcznego przyciskami

Parametry te dostępne są do edycji w pliku *focuser.ino* w pobranym pliku *Arduino sketch* (patrz sekcja powyżej) i należy je zaktualizować jeśli stosujemy własne ich wartości.

Sterowanie mikrokrokowe

Urządzenie domyślnie pracuje w trybie półkrokowym. Jest możliwa modyfikacja urządzenia i zastosowanie dodatkowego kontrolera który umożliwi sterowanie mikrokrokowe jeśli potrzebna jest większa precyzja ustawiania ostrości. Modyfikacja polega na wylutowaniu z płytki sterownika L298, umieszczeniu w odpowiednich otworach pinów i podpięcie do nich dodatkowego kontrolera ze sterowaniem mikrokrokowym na przykład opartego na popularnym układzie A4988 albo TB6560:





Powyżej po lewej mikro moduł z układem A4988 umożliwiający sterowanie z dokładnością do 1/16 kroku z maksymalnym prądem 1A. Po prawej moduł z układem TB6560 dający taką samą dokładność, ale więcej możliwości konfiguracji oraz większy prąd maksymalny (do 3A). Moduły takie można zakupić na allegro lub w sklepach elektronicznych w cenie około 30-50zł. Są również dostępne gotowe moduły sterujące z większą dokładnością (1/64 albo nawet 1/128 kroku) ale ich ceny są mało atrakcyjne.
W celu podłączenia modułu konieczne jest poprowadzenie 9 dodatkowych przewodów między układem focusera oraz dodatkowym kontrolerem:

1. 2 przewody doprowadzające zasilanie
2. Sygnał CLK taktujący sterownik (Arduino pin A4)
3. Sygnał DIR wskazujący kierunek obrotu (Arduino pin A3)
4. Sygnał ENABLE kluczujący zasilanie silnika (Arduino pin D6)
5. 4 przewody zasilające silnik krokowy

Ostatnią czynnością będzie zmiana konfiguracji silnika krokowego w kodzie Arduino:

```
AccelStepper stepper = AccelStepper(AccelStepper::DRIVER, A4, A3)
```