Plug N Play

Focus Controllers

V2.0.1

compiled by Todd Benko, 2020

Table of Contents

PREFACE

Disclaimer

This project is released into the public domain as is where is, with no obligation or responsibility accepted on the part of the author, for any mishaps or failures caused by this product or use of this product. Users intending to use this project or code do so at their own risk and usage of product and code is deemed to be acceptance of those risks. The author(s) accept no responsibility to damage caused to any equipment or goods or self by using the ideas, schematics and code associated with this project, or loss of income or all other losses that may be incurred. No warranty is offered or implied.

Copyright Restrictions

The schematic, firmware and other code and ideas are released into the public domain. Users are free to implement these for their personal use, but may NOT sell projects based on (or derived from) this project for commercial gain without express written permission granted from the author(s).

Firmware License Restrictions

If you copy or write new code based on the code in these files you must include a link to these files AND you must include references to the author(s) of this code. ***Firmware is released subject to the following restrictions***

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), the rights to use, copy, modify, merge, publish, distribute, sublicense copies of the Software (from the authors), **subject** to the following conditions:

1. The above copyright notice shall be included in all copies or substantial portions of this Software.

2. The software may NOT be sold or charged for in any way, either as part of a package, license fee, annual charges or premiums and shall include the copyright notice above.

3. The source-code to any software MUST be made available in full source-code form (including modified source files) and free to download without payment and include the copyright notice above

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

INTRODUCTION

Overview

An Arduino-based focus controller named "PNP-Focus". As the name implies, it is to be as easy as "Plug and Play" with little or no soldering. No hassle with firmware upload.
You simply 1) buy the readily available components, 2) stack them, and 3) upload the firmware.

The PNP-Focus is just the controller. You still need to mount a stepper motor on your telescope connected to this controller. With motorized focuser you get the higher resolution since a stepper motor may have 2000-6000 steps per revolution. The higher resolution is crucial for imaging. You have the option to control the focuser locally or remotely from the Personal computer (PC). When you do imaging, you control your devices via a PC and software. PC based then leads to automated focusing.

Features in V2

- Manual control.
- LCD display (Optional)
- Temperature sensor.
- Local manual motor control using automatic increasing speed changes.
- Independent motor speeds for local button focus position and ASCOM commanded position
- Stepper motor output power down after a positional move to prevent motor from heating due to quiescent current.
- Ability to drive Uni-Polar or Bi-Polar stepper motors.
- Full step and half step motor movements.
- Switching between full step and half step automatically converts the position counter to new step count and correct transition step code.
- Ability to save position and step mode to non-volatile memory and recall on power-up to restore unit to previously used configuration
- Compatible with Moonlite DRO ASCOM driver with the following compatibility features:
    - Full selection and operation of moonlight DRO ASCOM step speeds using 16, 32, 64, 125, 250 steps/second.
    - Temperature sensor for temperature compensation by ASCOM driver. Temperature sensor moved to pin A2.
    - ASCOM connection locks mode to driver selected Full step or Half step mode.
    - ASCOM stop or disconnect records stepper position and step mode to non-volatile memory.
- Additional features added when operating in ASCOM connection which are not provided by Moonlite Technologies DRO driver:
    - Backlash compensation for outward movement to compensate for gravity and to ensure consistent approach to focus position in either direction.

    - ASCOM moves use soft start with speeds increasing up to the selected driver step speed setting.
    - ASCOM move command have reducing speed as target position is approached to to ensure positional accuracy and repeatability.
- Compatible with commercially available focus drives using stepper motors
- Works with the popular stepper motors:
    - ABS3008-002 (12V) RoboFocus Stepper Motor
    - LSG35012F76P (12V) Moonlite Telescope Hi-Resolution Stepper Motor
    - 28BYJ-48 (5V and 12V)
    - 17HS13-0404S-PG27 (12V)

Exceptions

- PNP-Focus units will not drive DC or AC servo motors used in some focuser drive systems.

Caveat

The goal behind PNP-Focus was to use off the shelf components, plug them together without the need to perform any soldering. The primary components can be assembled within the design scope. The real issue could be the components you select to assemble your project may indeed require some assembly or soldering activities. Many of the components from suppliers may in fact not have the header wires installed. Also depending on your housing your interface connections may requires some assembly or soldering. Interface components such as power wires, motor connectors may require soldering assembly depending on components that you source.

Revision Notes:

Version 2.0.1: Fix a bug, where runs longer than 30 seconds would cause motor to halt abruptly and then start back up. Issue was caused by temperature sample taken at 30 second interval, which interrupted the motor move sequence. Also removed dependency on requiring steps per revolution. Original stepper motor library based speed decisions on RPM. During discussion on forum thread about what speed to used on belt drive design discussion, I realized that setting speed to Ascom driver speed selection differed with use of RPM speed calculated results. As result converted stepper motor to strictly use steps/second velocity to determine step timing interval rather than RPM and eliminated need to enter any motor step information. Half step motor drive problem had been reported, confirmed the motor drive sequence did not fire in the same sequence as documented in manual for the 28BYJ48 motor. Added further documentation to program and to this document about LED signals and corrected firing sequence. Concern is that different stepper motors may require different firing order sequence for half step operation. Hopefully half step order control is not manufacture dependant. Also changed

the start start and stop operations to a fully ramped speed control rather than a 4 step speed control change to provide a much smoother transition from starting and target position changes.

Version 2: As a result of not being able to have the firmware recognized by the Moonlight DRO focus ASCOM driver, I decided to take on the task of revamping the whole Arduino firmware and interface was developed. I didn't like the way the code was structured. The previous code had so many limitations and did not fully implement Moonlite DRO ascom driver features.

In testing the new version, it became apparent that sometimes the PNP-Focus would be detected by the Moonlite DRO ASCOM driver as a 2 driver board version. Code was installed to respond for 2 driver version. Later it was determined that it might not be needed because of the specified version response to the driver and was commented out. One of the first tasks was elimination of the delay command for use in many of the areas which was slowing down the Arduino execution. Another goal was to make the code adaptable so the same code could be used to load both the standalone PNP-Focus board and the PNP-Focus stick rather than having separate versions. In addition it would allow the PNP-Focus board to run headless without the LCD1602 Shield if a person decided not to install. Another of the key features was ensure the unit was fully compliant with how the Moonlite DRO driver worked. I discovered that while some commands were available in the driver listing, Moonlite Telescopes didn't necessarily implement the function in the device and firmware. Key elements included speed control which matched the driver, setting and reporting proper speed control. The original driver used a fixed speed that was not controlled by the driver. In managing the interface to the Moonlite commands in the DRO driver, a Moonlite command library was added to the project. Further enhancements developed included configuring the data display to display more useful information and expanded functionality. Expanding but also simplifying the display button control with more brightness levels. The addition of ramping manual speed control which eliminated the need to manually change speed. The new code totally made use of the stepper timing for specific speed based on steps per second rather than simply using time delays. The ardumoto library was fixed properly so the correct speed would be based on the number of steps per rotation of the stepper motor. In addition when a move was completed, the motor drive outputs would be disabled, ensuring the motors didn't need to heat up when not moving. Full step and half step stepper mode was implemented with corrections again to library code that was previously conceived. Also included the ability to provide backlash compensation for focus rack-out position changes. Also it was recognized that the higher Moonlite speed levels have been known to cause focus tube slippage with heavy photography equipment loads on the focus tube. To address this problem, both soft start position change and slower target position settling speeds were added. Now higher ASCOM speed settings can now be used without compromising positional accuracy and repeatability. In essence, the firmware was totally re-written and restructured. The main loop now contains only a small repeatable core set of element actions removing all the uncessary delays to the firmware cycle time. Because there are so many significant code and functional changes, this version is re-branded as PNP-Focus V2.

Version 1 variations: These versions had their simple firmware configuration based on work started by Orly Andico and the cloudy nights thread started by Anat Ruangrassamee. There have since been 5 other firmware versions spawned where people have adapted the code to their specific purpose and shared. The problem with many of these older versions is that many would not be detected by the new Moonlite DRO focus ASCOM driver.

Hardware Variations

There exist two design hardware versions. Both version offer computer focus control through the Moonlight DRO Ascom driver. There is the full size standalone PNP-Focus Controller offers USB and local button focus control. The PNP-Focus Stick only provides computer focus control via the USB port.

Standalone PNP-Focus Controller

The Standalone focus controller is intended to provide a user with direct control of the focus motor without any need for software or computer. The designed is based on the Arduino Leonardo board, using a plug in LCD1602 display shield as the user interface. The display shield offers a 2 line x 16 character display for status information. The shield includes 6 tactile buttons to provide manual focuser control without any computer connection. Telescopes with an electronic focuser can be used without the need to connect to any computers nor the need to remove the focus motor for simple astronomy usage.

The standalone focus control still provides full computer control when connected using the arduino USB connection and the moonlite DRO ASCOM driver. The same design can run headless, without the LCD1602 display shield installed. The variation would provide using only computer control in the similar form factor.

Required Components

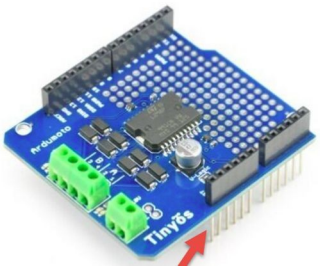
1) Arduino Leonardo ATMega32U4 Controller

The Leonardo board with the ATMega32U4 chip is the board that was used for this build. Technically the pin layout for the Arduino and the Ardumoto board would line up for use with other Arduino controller boards. For simplicity sake just use the Arduiono Leonardo. The code may work with other Arduino board with the same footprint because you use the Arduino integrated development environment (IDE) program to load the firmware. We leave that choice up to those who want to play rather than just using what was chosen for the PNP focus platform.
search “Arduino Leonardo” on by Arduino or keyestudio on Ebay, Amazon or Aliexpress

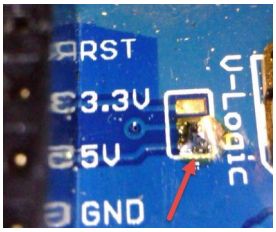
Keyestudio Leonardo, Ebay,

2) Ardumoto L298P Motor Drive Board

Ardumoto L298P motor drive board can be purchased with different pin assignments
The standard for this programs uses



D3 -A motor PWM drive
D11-B motor PWM Drive
D12-A Dir
D13-B Dir
The L298P board is capable of driving up to 2A current per motor connection. To make things easy just ensure the board you purchase uses Pin 3, 11, 12, 13 for driving the motor. Fortunately, most of them are. Also ensure the socket with leads come with kit so stacking above and below board is available. Search “ardumoto L298P on Ebay, Amazon or AliExpress. Do not confuse this with Ardumoto version 2.3 boards using L293D chips. Their control is totally different and not compatible.



Many of the ardumoto boards come with surface mount pad instead of jumper pins to select the logic voltage between 5Volts and 3.3Volts. The Leonardo requires using 5 Volt logic to be selected. **Ensure that you place a solder bridge on the 5V selection pads to set the correct logic voltage.** A small single strand of wire helps to make this bridge. Make sure there is no connection to the 3.3V pad. If it is not selected, move commands will just cause the stepper to vibrate or move really slowly when issuing a move command. This issue was most people’s problem in building their PNP-Focus controller.

Keyestudio Ardumoto Shield , Ebay

3) 1602 LCD and Keypad Shield

Search “1602 LCD and keypad shield” on DFRobot, Ebay, Amazon. There are various 1602 keyboard shields available and most should work. LCD shield for this project uses Pins D4-D10 and A0. The differences are usually button sizes, additional interface pin and how the LCD contrast is adjusted.

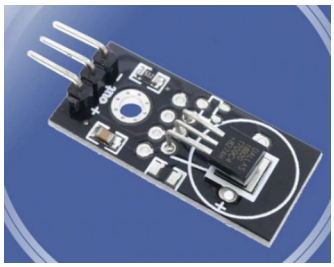
1602 Shield for this project uses the following pin assignments
D4-LCD D4
D5-LCD D5
D6-LCD D6
D7-LCD D7
D8-LCD Reset
D9-LCD Enable
D10-LCD BackLight brightness adjustment
A0 – is button decoding.
Button readings have two different analog ranges depending on which version is purchased. Need to select using comments // which pin assignment works for your LCD shield in read_LCD_buttons() function.

Make sure the one you select works for the case you plan to use and how you plan to implement the temperature sensor. The DFRobot has longer tactile button caps and has pins to make temperature sensor connection to the LCD shield through the top of the case using solderless pin connections. Others have lower tacktile buttons and there is no need for any pins on top surface so you want pins on the bottom for stacking onto the ardumoto shield. If you do not want to make a different temperature connection through the LCD top surface, use the breakout area of the ardumoto to wire in your temperature sensor connections using soldered wire connections. You could use an internal temperature sensor, an external one through the motor drive cable or a seperate standalone plugin that uses a headphone jack connection which could disconnect an internal temperature sensor connection.

Keyestudion LCD1602 Shield , Amazon , Ebay, AliExpress

4) DS18B20 Temperature Sensor and Interface

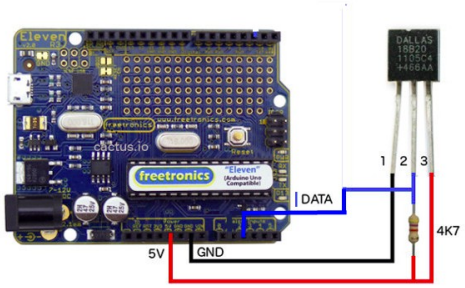
There are so many different ways to connect the DS18B20 temperature Sensor. For the PNP-Focus the sensor data pin will connect to the Analog A2 Arduino pin. The original PNP-Focus used pin A2, however the PNP-Focus stick has a conflict with nano-motorshield. Therefore this version and firmware code is looking for temperature probe data on Analog pin A2. The interface requires that the data pin be pulled up to 5V with a pull up resistor either by a non-soldered interface board or by your own wiring application. It basically depends on your assembly comfort level. The temperature sensor is only used for temperature compensation focus position moves. The temperature must reflect the temperature change of your telescope and it will vary unit design variances. There are assembled boards which have resistor and indicator diodes already installed that might be able to placed on outside of housing and just require running wires to the Arduino pins.



Another approach would use a waterproof external temperature probe which would be attached to the shell of the telescope to measure the temperature changes. Another approach might be to install the temperature sensor or module board in the dB9 connector which attaches to the focuser motor. All approaches may have their differences and any application of temperature compensation is based on an observed response based on either temperature sensor location.

Using either method you need to observe the focus changes which occur with your telescope prior to applying a temperature compensate rate.

The moonlite focuser lists commands as though temperature compensation would be applied by the focus controller. Moonlight Telescopes Ltd has been determined that temperature compensation is better managed by the ascom driver, rather than the hardware interface. It has been validated that the temperature compensation commands listed in the hardware command interface are not issued by the driver when temperature



compensation is selected. As a result temperature compensation is not enabled when operating the as a standalone controller. The fact that manual focus control is more important to adapt to different users in visual astronomy or other standalone astronomy usage rather than the benefits offered by temperature compensation.

Keyestudio DS18B20 , Amazon , AliExpress , Voltateck

5) Interface Components

You need a variety of interface components to power and connect your PnP Standalone Focus Controller to your focus motor assembly.

Vin Power connection:

12V power cable to power the Arduino and Stepper motor. The ardumoto shield will not drive the stepper motor with only USB power applied. The ardumoto shield requires that Vin power be supplied at the voltage required to drive the stepper motor. Typically recommend using 12V stepper motor. The Vin can be connected using the barrel connection of the Leonardo or the terminal strip on the ardumoto. For the PNP-Focus Stick it must be connected to the terminal strip. Solderless barrel connectors are available from various sources.
Amazon , Adafruit , Ebay

- USB micro cable to program the arduino and to drive unit with PC using ASCOM driver. The assembly of the PNP-Focus does not provide good accessibilty to the USB interface. One solution is to use a USB patch cord with a panel mount usb connector. This way you do not need a specific as built case to house the PNP-Focus controller. To conserve space, recommend looking for the 90 right or left usb patch cords.
AliExpress

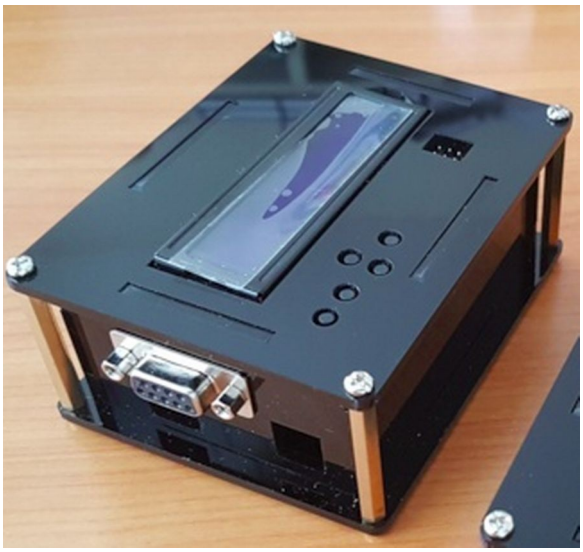
- A cable interface from the controller to the motor. Many focus motors use a dB9 connection. A dB9 male-female cable would be used if a dB9 connector was installed on the controller housing
- option using an Ethernet cable and installing a female Ethernet jack on the controller housing and then a RJ45-dB9 adapter at the motor drive connection.

Standalone Computer Only Configuration Option

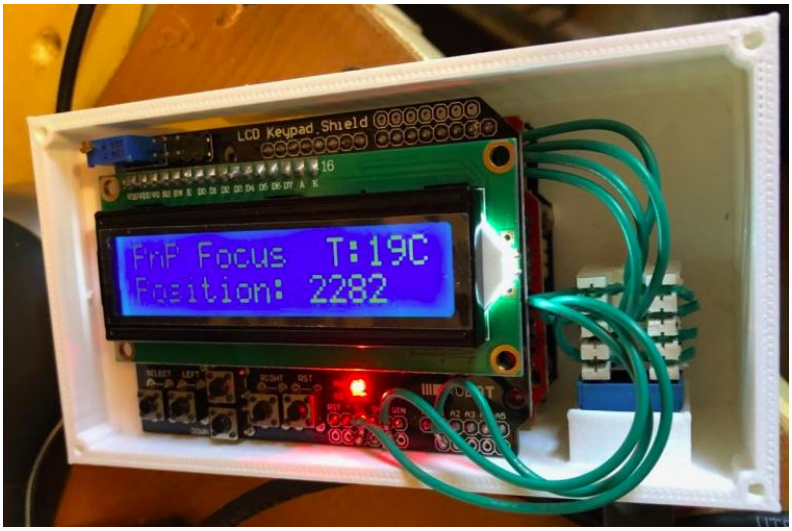
If you like standalone hardware and case you plan to use and only want to drive it from a computer ASCOM driver, you can just omit the installation of the LCD1602 display and keyboard shield to produce only computer access focus control. For this configuration, you need to adjust the user configuration in the arduino firmware code to comment out LCD1602 display and keyboard shield as it is not installed.

Case and Housing Options

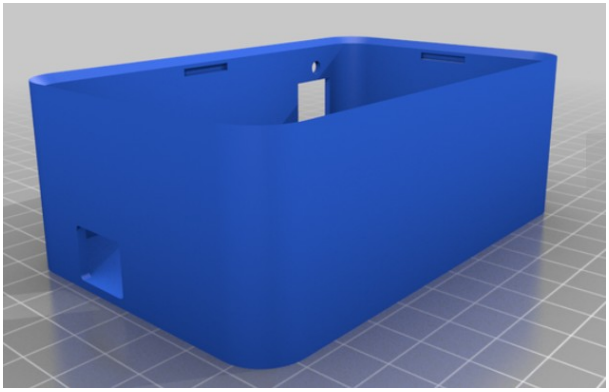
Standalone PNP-Focus Controller builders have found various methods to enclose their controller boards. Anat used flat laser cut plexiglass material to create a jointed box which is fastened together with 4 screws. The laser cut dxf files are included in the zipped file assembly.

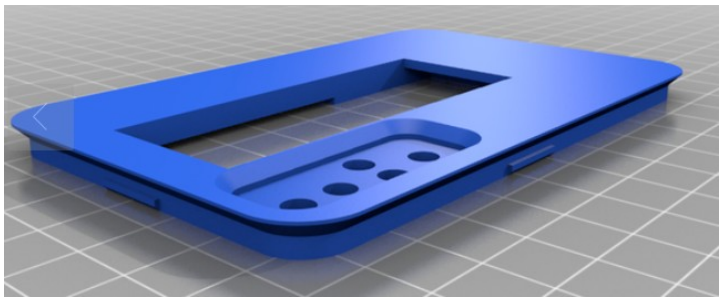


Mr. Tom K. created a 3D printed case and lid to enclose his standalone PNP-Focus controller.



The following box was created by RJ2K19 and uploaded to [PNP-Focus Box on Things in the Universe]. In this build, a RJ45 connector is installed to use Ethernet cable to interface to the motor housing.



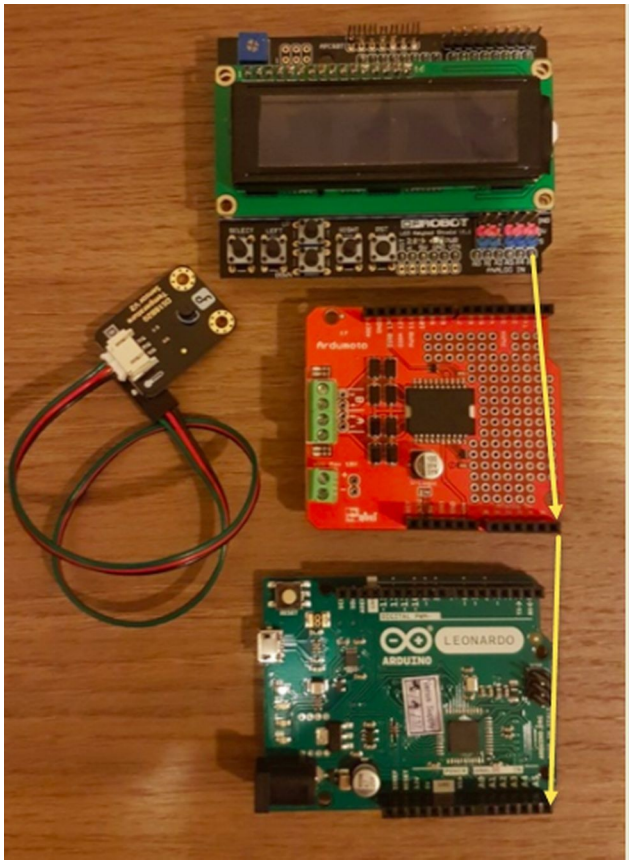


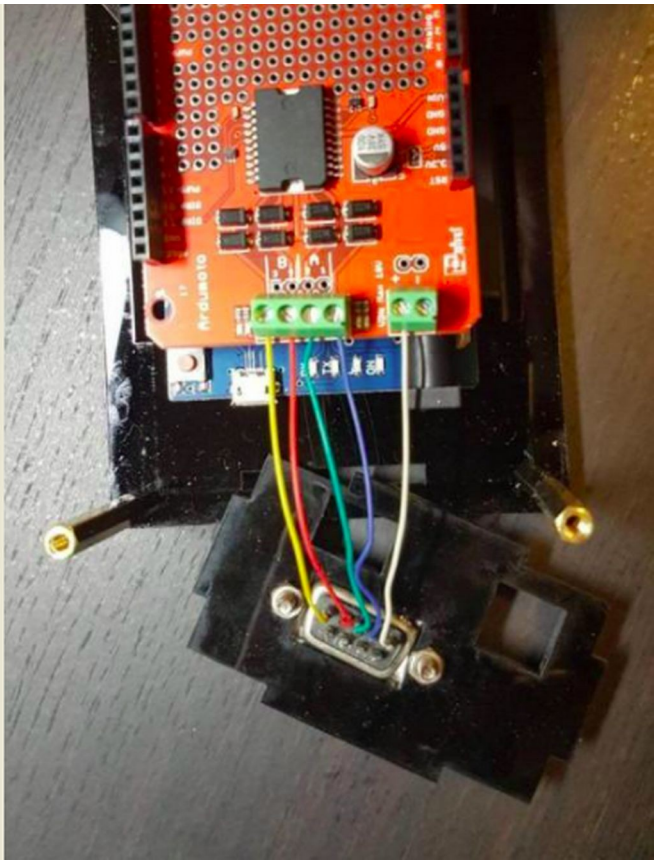
Jem45472 found an outdoor electric plastic outlet box as a suitable enclosure for his build

I used a standard plastic Hammond case 120mm x 65mm x 40mm and just carved out the holes and slots for the interface components.

Assembly Notes:

1. Many of the Arduino and shields do not come assembled with the stacking pins installed. The ardumoto board is in the middle of the stack so it must have stack-able headers installed.



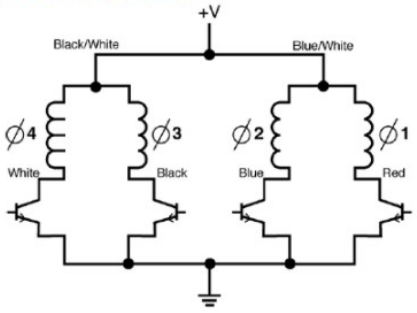


When you stack the boards together, you want to ensure pins are aligned such that all digital and Analog I/O pins align on each stacked board.

2. Wiring the stepper motor is a bit confusing. It doesn't help when board silkscreen numbers or user documentation doesn't match with assembled components. For instance Moonlite hi-res focuser is documented as the following pin and wire colour assignment.

AH DB9 Pin	MoonLite ROBO Focus	Hurst LSG35012F76P
1	Coil 1 +	Black
2	Coil 1 -	White
3	Coil 2 +	Blue
4	Coil 2 -	Red
5	+12V	Blue/White Black/White
6		Open
7		Open
8		Open
9		Open

LSG35 Geared Diagrams:



	Ø4 White	Ø3 Black	Ø2 Blue	Ø1 Red	
CCW ROTATION	1	0	1	0	CW ROTATION
	1	0	0	1	
	0	1	0	1	
	0	1	1	0	

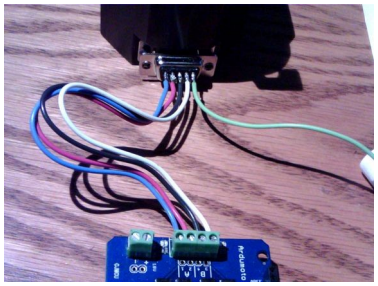
1 = ON, 0 = OFF

SWITCHING SEQUENCE

However, disassembly of a Moonlite hi-res focus motor on a Newt telescope found Blue on the dB9-pin 1, Red on pin 2, Black on pin 3 and white on pin 4 and the blue/White/ Black/White wires both connected to pin 5. This is a significant difference and if we plot out true table in firing the same order but in a different wire order we find the following:

	A		B			A		B	
	WHT	BLK	BLU	RED		BLU	RED	WHT	BLK
CCW ↑	1	0	1	0	CW ↓	1	0	1	0
	1	0	0	1		1	0	0	1
	0	1	0	1		0	1	0	1
	0	1	1	0		0	1	1	0
	1	0	1	0		1	0	1	0
	1	0	0	1		1	0	0	1
	0	1	0	1		0	1	0	1
	0	1	1	0		0	1	1	0

The left side is the documented order and the right is same order firing but wired in reverse. If we look for the same exact logic sequence of the right table in the left, we find the firing order produces an opposite rotational direction.



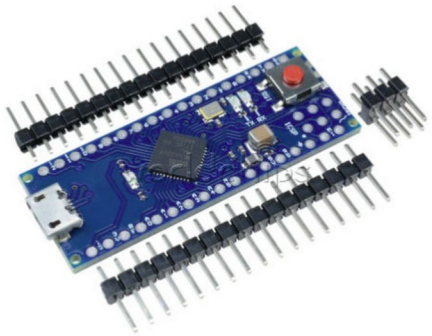
This is significant to your installation of the stepper motor to the telescope. The focus controller is designed where increasing the step count produces an outward focus movement. If you find that your motor is running inward with increasing motor steps, all you need to do is swap the wires in pair A and B output on the Motor shield or on your stepper motor connector. Voila, the focus runs in the correct direction. More details about the specific wiring connection are provided later in the Operating Principals section.

PNP-Focus Stick Controller

The desire to have a smaller focus controller which can only be controlled by computer appeals to many users. With computer access control, there is no need for the display and keyboard shield. You could simply not install the LCD1602 shield and use the stand alone board configuration. However there is a smaller Arduino unit which can make a smaller focus controller unit. The PNP-Focus Stick uses an Arduino MICRO board with a nano motor shield which fits the smaller pin layout configuration.

Components required:

1) Arduino Micro ATmega32u4 Controller:



Ensure you select the Arduino Micro ATmega32u4 5V 16MHZ 100% for Arduino Micro Replace pro mini and not the Mini or Mini Pro versions as those have a different pin footprint which does not match the motor shield. When assembling this board you need to ensure the pins are installed pointing downwards so it will plug into the and above the NANO motor shield.

Ebay,

2) NANO Motor Shield:

Nano motor shield board is believed to be standardized with the same pin assignments used in the program. When searching, just confirm the following pin assignments.

The standard for this programs uses
D3 -A motor PWM drive
D11-B motor PWM Drive
D12-A Dir
D13-B Dir
To make things easy just ensure the board you purchase uses Pin 3, 11, 12, 13 for driving the motor.

The shield can supply 2 amperes per channel, for a total of 4 amperes maximum.

Input and Output

This shield has two separate channels, called A and B, that each use 4 of the For Arduino pins to drive or sense the motor. In total there are 8 pins in use on this shield. You can use each channel separately to

drive two DC motors or combine them to drive one Bi-Polar stepper motor. PNP-Focus only uses the code to drive the stepper motor operation and not the DC motor operation. The shield's pins, divided by channel are shown in the table below:

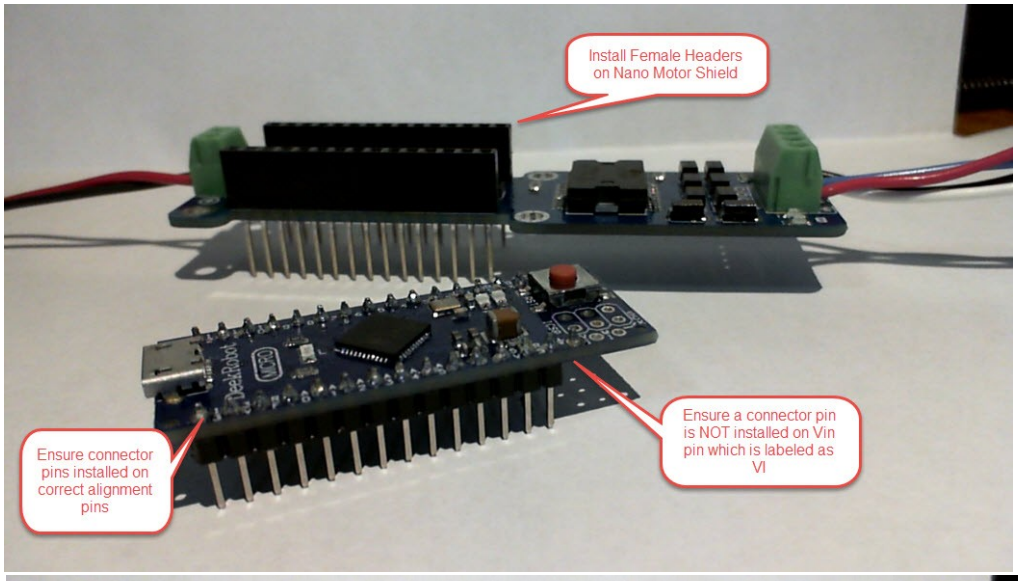
Function	pins per Ch. A	pins per Ch. B
Direction	D12	D13
PWM	D3	D11
Brake	D9	D8
Current Sensing	A0	A1

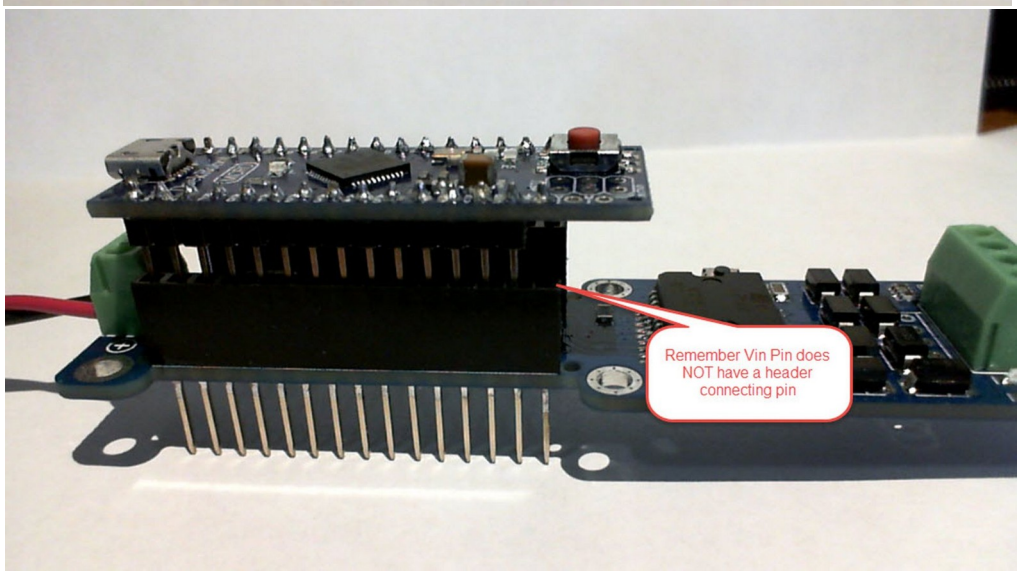
AliExpress , Amazon , EBay

3) Temperature: Can use the same resources as the PNP-Focus stanalone version. Soldering or wirewrap required to pins into place for proper temperature placement. For the temperature you need the same connections as in the standalone version to +5V, Ground (GND) and the centre data pin connected to Analog 2 (A2) pin. It is conceivable that the PNP-Focus Stick would be used to build a complete focus assembly with the controller and motor in one housing and attached directly to the telescope. In this situation you really need to ensure the temperature probe gets outside the housing. Both the stepper motor and the Arduino boards will produce heat inside the housing. That heat will throw off your focus set point because it will cause the focus to move after you have each use of the focus drive.

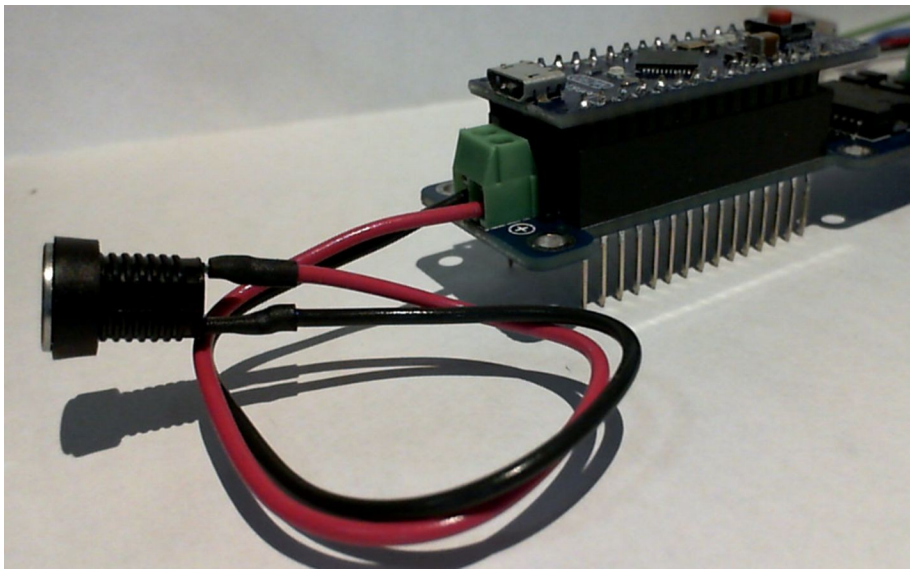
PNP-Focus Stick Assembly

1. Typically the Arduino Micro and the Nano motor shield are delivered without any header pins installed. You will want to install the Micro header pins downwards and you will need to install female header pins on the NANO motor shield. NOTE: Do not install a header pin on Vin pin on the Micro board. See the technical specification at end of the document about why the Vin connection is removed between the Micro board and the Nano Motor Shield for all applications.

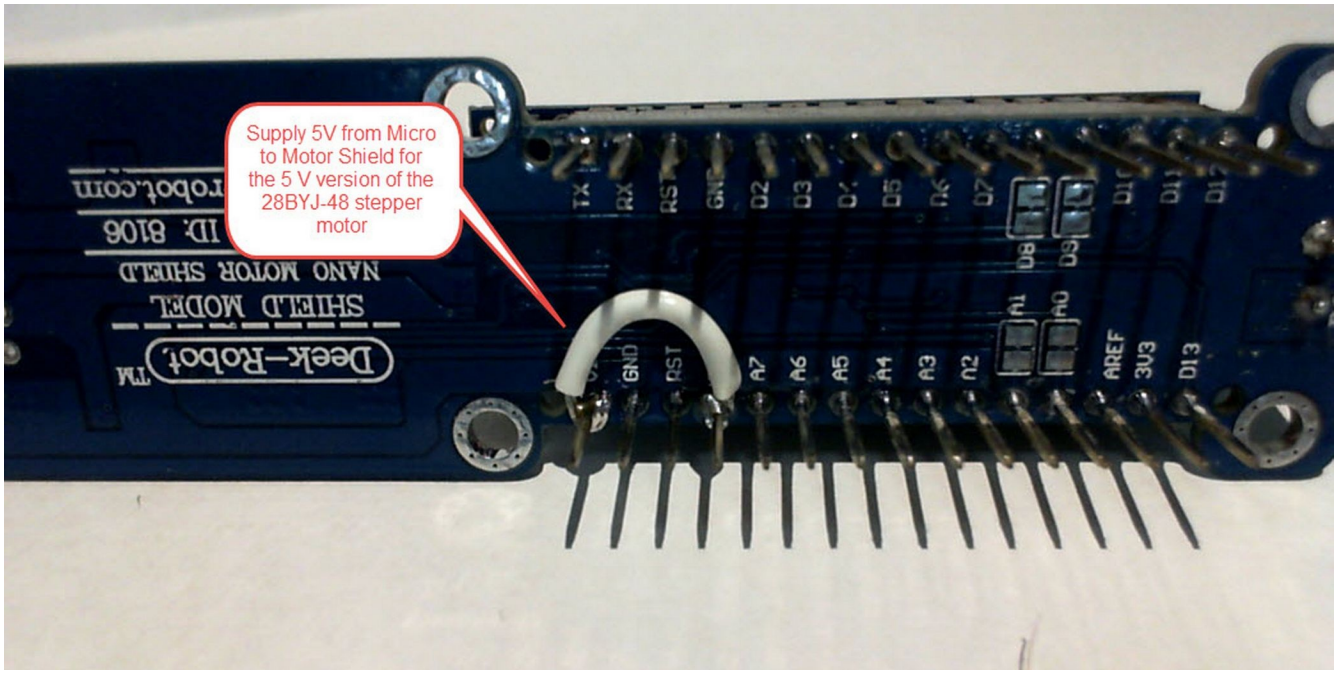




2. If using a stepper with voltage > 5V then connect a power supply connection to power the Stepper motor with the required voltage.

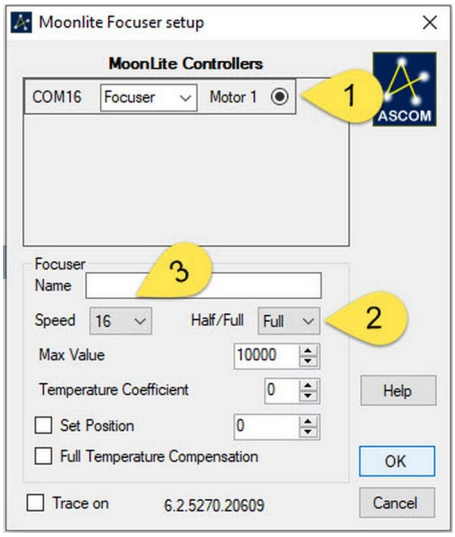


3. If you are intending to only use a 5V version of 28BYJ-48 Stepper motor then you need to supply a 5V supply to the motor shield board. One approach is the same as previous step where 5V would be supplied from external power cable. A simple approach is to use the 5V power from the USB connection to the MICRO. In this case you need to connect 5V pin to Vin pin on the bottom of the Nano Motor Shield.

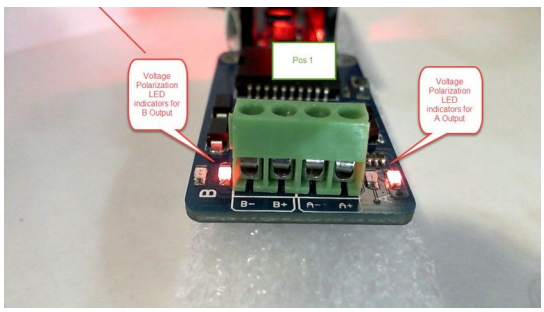


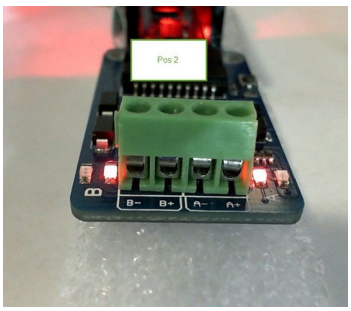
4. Don’t connect motor yet. Power up the stick with both USB and designated motor power connection and load the specific user configured firmware into the MICRO.

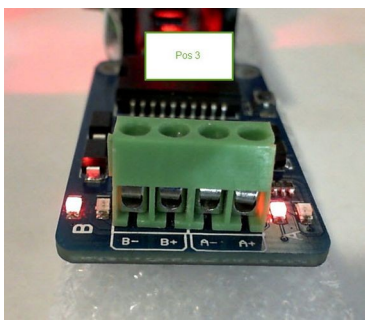
5. Launch a program which can connect to the PNP-Focus Stick with the Ascom Connection. Make sure to select only Full step operation, and for testing purpose set speed to 16 steps/second.

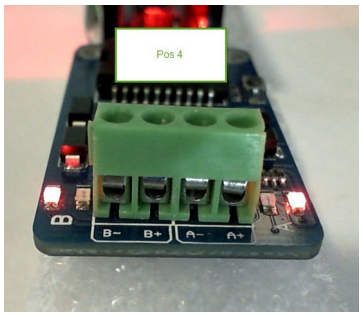


6. Connect to the focuser and move the focuser to step position 2. Observe the Output Voltage Polarity LED indicators on either side of the motor connection terminal block. In Full Step mode an LED on both A and B output should illuminate for each step. The LED should illuminate during the step move and after 3 seconds extinguish, indicating the power has been turned off to the stepper motor. Full step has a 4 step sequence so go ahead and step one at a time through position sequence 0-1-2-3-4-5-4-3-2-1-0 and observer the sequence should look something like the following.





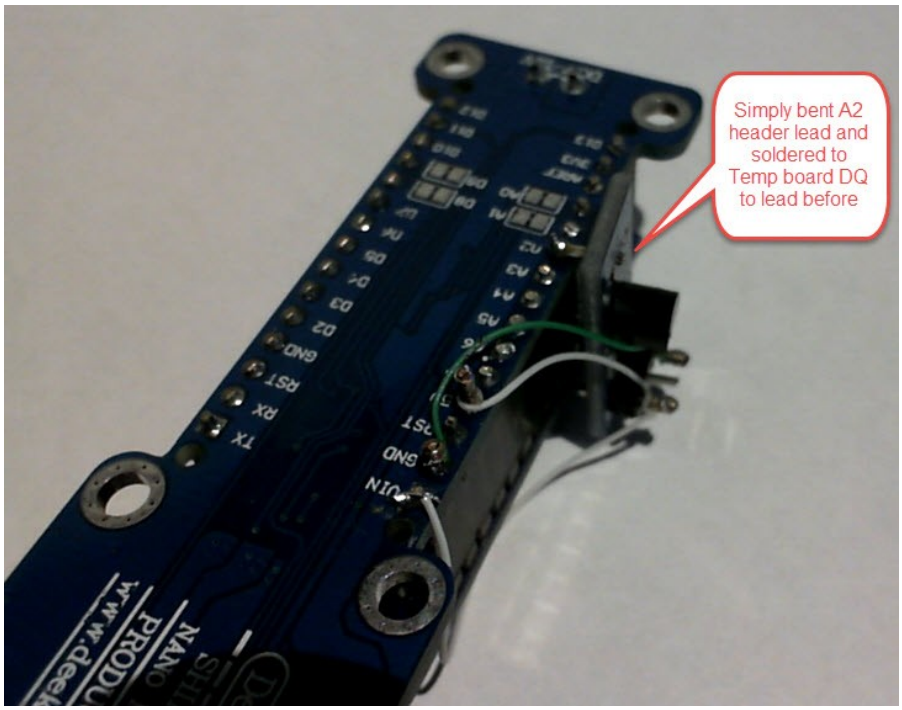




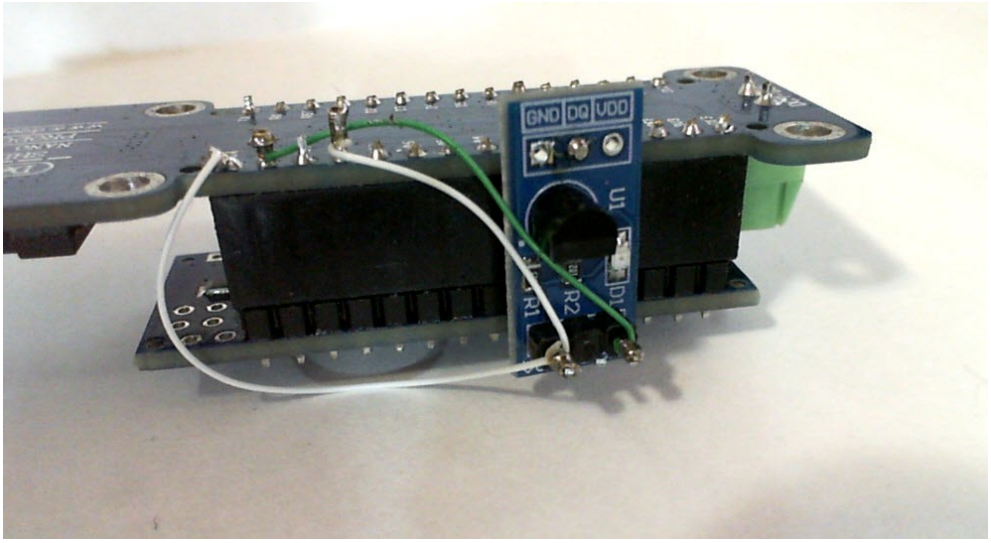
7. Now power things down and connect up the lead to the stepper motor. Refer to Operating Principles section on Motor Wiring connections.

8. Repeat tests again with motor connected to ensure proper stepping sequence. Once completed you can then use a larger step move to determine if you have the correct stepper motor rotation direction for an increased step position.

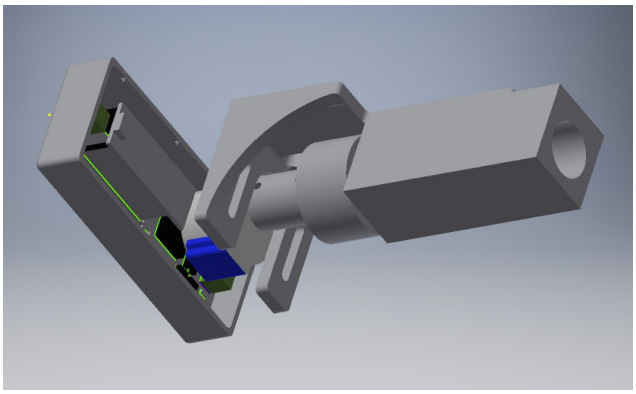
9. Temperature Option: Attaching a temperature probe is a bit more challenging if you desire. I chose to go with an internal temperature sensor. I simply bent the lead for pin A2 over at 90 degrees and soldered a simple temperature probe board on to the lead before trimming the lead. I then ran wire

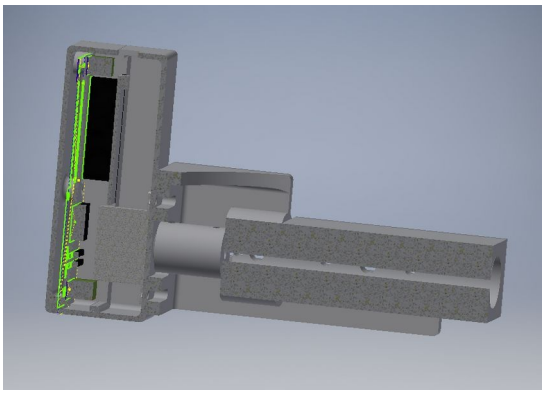


wrap wire to connect the 5V vcc and Ground connections to the other pins before trimming the leads. Its not a true telescope temperature indication, but it will work for my purpose. Here are picture of a possible installation option:



Option:
Here is my build of a complete electronic motorized focuser based on PNP-Focus Stick and the 5V version of the 28BYJ-48 stepper motor. https://youtu.be/u_0s6xRi2YM





Installation

Firmware Installation

The firmware with this version is user configurable and will work with both the Standalone PNP-Focus controller and the PNP-Focus Stick. The difference is that a user must modify the user definitions at the start of the arduino code to fit their stepper motor configuration and the installation board options.

1. Download and install the Arduino Integrated Desktop Editor (IDE) from the location:
   You only need to install the program on a computer which will be used to load the firmware into the Arduino board.

2. Download the project files for the Github repository from https://github.com/benkot/PnP-Focus on the same computer as the Arduino IDE. Recommended directory is Documents\Arduino and unzip the files. Descend into the directory for the PnPFocuser-V2.0 There are many different forks of the original version so you need to have the version which identifies PnPFocus-V2.0.

3. Double click on the file: PnPFocus-V2.0.ino to launch the Arduino and open the file in the IDE.

4. The front or top part of the firmware code contains user definitions which must be tailored to your configuration and operational choices.
   4.1. Motor operation can be in Full step or Half step mode. Many stepper motors do not appear to operate in a repeatable manner in Half step mode. However if desired to operate in Half Step mode, change the line in the firmware code:
   enableHalfStepMode = true; // false only enables only Full Step operating mode, true enables both Full step and Half step operating modes.

   4.2. Backlash Compensation: Some focus motors when racking out create a shift due to gravity on the weight of the equipment on the focuser or the play in the gearbox between the motor shaft and the output shaft. To compensate for this issue it is often desired to approach the focus position by first overshooting the outward move and then have a consistent inward approach to the target position. The backlash compensation will only apply to ASCOM driver moves. Any position moves less than 5 steps will not have backlash compensation applied to the outward position movement. Inward postion movement never have backlash applied due to the nature of the movement and effects of gravity. If using the Ascom driver temperature compensation features the temperature moves are small one step changes at a time and will not have backlash compensation applied. The DRO ASCOM driver does not have ability to set the backlash value. Therefore backlash value must be set in the user configuration area of the firmware code.
   unsigned short backlashValue=0; // increase this value to indicate how many steps your focus motor needs to overshoot to compensate for focus backlash. The focus

motor will rack out this number of steps extra and then back in the same number of steps to arrive at the target position.

Note: Some image capture software include their own backlash compensation control option in their program operation. If you want to use backlash compensation, ensure you only apply configuration values to either the PNP-Focus firmware or the capture software but NOT both at the same time. If you want to use image capture for backlash compensation, then ensure backlash value above is set to 0 in firmware user options.

4.3. The LCD1602 display Shield Configuration: If you have built the standalone PNP-Focus version you have a choice have the LCD display is installed or not. If you have built the PNP-Focus stick version, the LCD1602 configuration line must be configured as not being installed.
If the LCD1602 display shield is installed following line should be
#define LCD1602Shield
If not installed, the line requires comment characters (//) added so it looks like
//#define LCD1602Shield

4.4. Temperatures Units if using a DS18B20 temperature sensor: The temperature in both the standalone display and the ASCOM driver, values can be either Celsius or Fahrenheit. Change the line in the firmware code to your temperature unit choice.
bool displayFahrenheit = false; // false = temperatures in Celsius., true = temperatures in Fahrenheit. If set to true, both display values and temperature compensation must use temperature values in Fahrenheit.

4.5. All standard user configuration is completed. The firmware code will now program either the Standalone PNP-Focus controller or the PNP-Focus Stick hardware configuration.

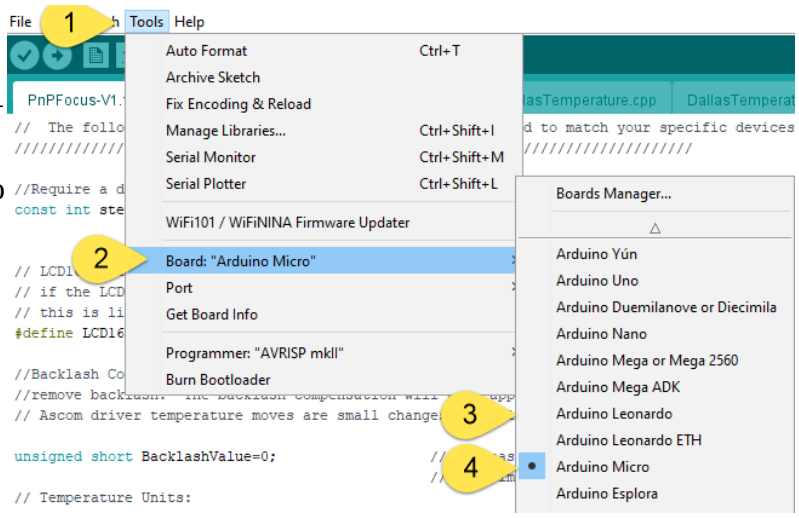
4.6. If installing the LCD1602, program the unit and validate the tactile switches function properly. Different manufactures of LC1602 boards may use different resister combinations for the switch measured values. You may need to change the commented out lines in the firmware “read_buttons” function if you encounter problems with your button action.

5. Assemble your PNP-Focus controller boards
   5.1. Arduino leonardo (bottom), Ardumoto shield (middle), LCD1602 shield(top) {Standalone}
   5.2. Arduino leonado (bottom), Ardumoto shield (top) {Standalone headless}
   5.3. Micro motor shield, (bottom), Arduino micro (top) {Focus Stick}

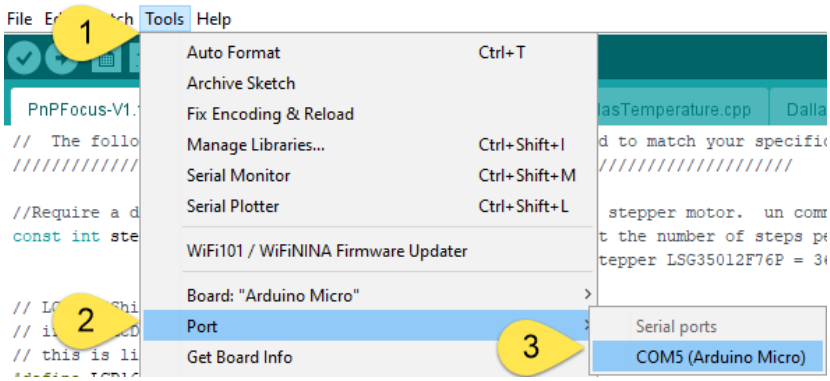
6. Connect your computer to the Arduino using a USB cable. The Arduino should self register as a USB comm port.

7. Use the Arduino IDE with the user modifications to compile the firmware code and load the arduino with the following steps:

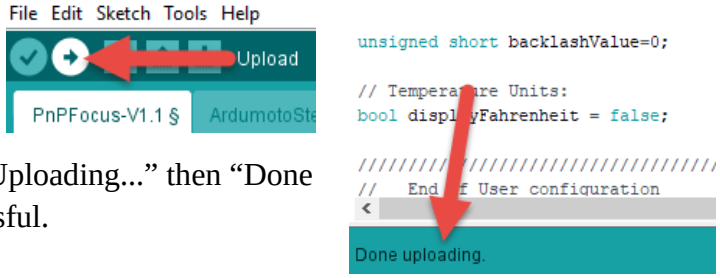
7.1. Set the board type in the IDE interface using menus Tools – Board – Select either the Arduino Leonardo or Arduino Micro as required for your Arduino board.



7.2. Set the communication port to which your computer is connected to the Arduino. Use menus Tools – Ports Select the appropriate com port which is identified as your Arduino.



7.3. Compile and load the firmware onto the Arduino from the IDE using the upload button on the tools ribbon. The bottom IDE status bar will display "Compiling Sketch", then "Uploading..." then "Done uploading" if the process was successful.



7.4. If you have the standalone PNP-Focus with the LCD1602 shield, try the buttons for proper functionality. If buttons respond different than expected, then there is a different set of threshold values to use in the read_buttons function. In the following example, the newer board thresholds are commented out. Add comment blocks to older board section and remove comment block from new board section. Reload firmware and try again.

```
// For newer boards may use these thresholds
/*
if (adc_key_in < 50)   return btnRIGHT;
if (adc_key_in < 250)  return btnUP;
```

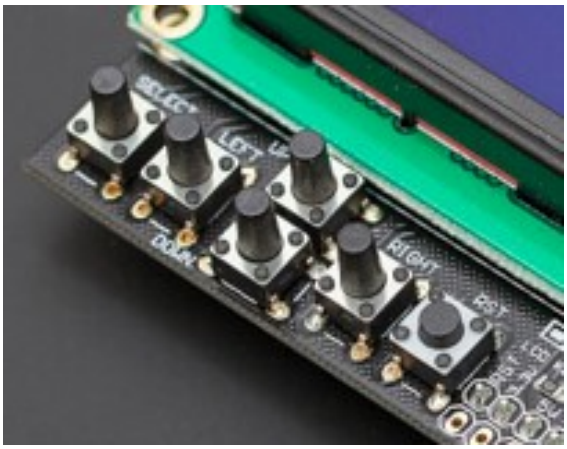
```
  if (adc_key_in < 450)  return btnDOWN;
  if (adc_key_in < 650)  return btnLEFT;
  if (adc_key_in < 850)  return btnSELECT;
  */
 // Older boards or clones may use these threshold:

   if (adc_key_in < 50)   return btnRIGHT;//increase value
   if (adc_key_in < 195)  return btnUP;   //screen ON/OFF
   if (adc_key_in < 380)  return btnDOWN; //No Activity
   if (adc_key_in < 555)  return btnLEFT; //Decrease value
   if (adc_key_in < 790)  return btnSELECT; //Stop and save

  // use comment block /*     */ to comment the whole group which is not
applicable to your LCD1602 shield
```

7.5. Recommend that you save the PnPfocus file with your specific configuration changes. This way it will be ready to use any time in the future for loading additional focus controllers for different telescope configurations.

User Interface Operations

Standalone PNP-Focus Button Controller

The PNP-Focus controller with the LCD1602 has 6 buttons for local user control. The buttons are typically labelled on the shield from left to right as Select, Left, Up, Down, Right and Reset.



Select Button: Save and Stop button. If the focuser is moving this will abort the move and will also save the current position to non-volatile memory. Typically this is used at the end of an observing session prior to shutdown if the focuser is not racked all the way in. The current position will be saved. The next time the focuser is powered up it will start with the current position as the last save value.

Left Button: Will decrease the focus motor position count which should rack the focuser inward if the focus motor is installed and wired correctly. Holding the left button will move the focus motor in an increasing speed manner. Movement will start of slowly at 2 steps/second. The speed will increase at 5 second intervals using 10, 50, 100 and 200 steps/second increments. The ramping speed control provides precise small step position response for accurate focus placement, yet still provides easy access to rapid focus position changes.

Right Button: Will increase the focus motor position count which should rack the focuser outward if the focus motor is installed and wired correctly. Holding the Right button will move the focus motor in an increasing speed manner. Movement will start of slowly at 2 steps/second. The speed will increase at 5 second intervals using 10, 50, 100 and 200 steps/second increments. The ramping speed control provides precise small step position response for accurate focus placement, yet still provides easy access to rapid focus position changes.

Up Button: Display brightness control. The display brightness cycles through 5 different brightness levels to enable unit to be totally off for night operation to very bright for daylight operation. Pressing the up button will cycle the brightness through the 5 brightness levels of Off (0), Very Dim(1), Medium brightness (5), bright (20) and Very Bright (200).

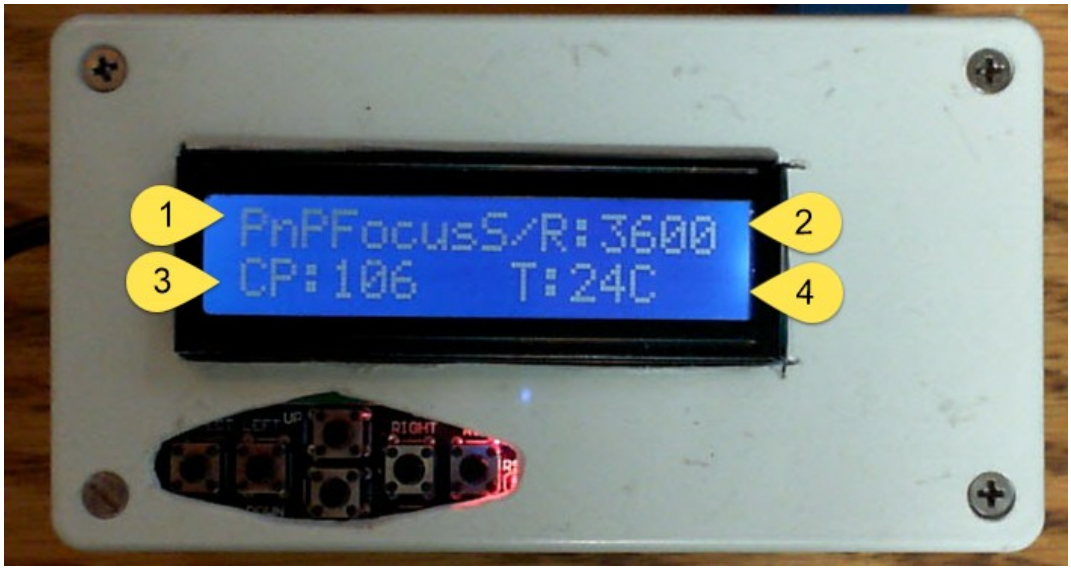
Down Button: Changes the stepper mode to either full step or half step mode. When the ASCOM driver is connected and sets the step mode, the down button is locked out from making changes. When

step mode is changed the current counter position will change to represent the current value in full steps or half steps.

Reset Button: When pressed will reset the Arduino to a power up state. The Arduino will stop all functions and reload the firmware from memory and restart as if power was just applied.

Standalone PNP-Focus Status Display

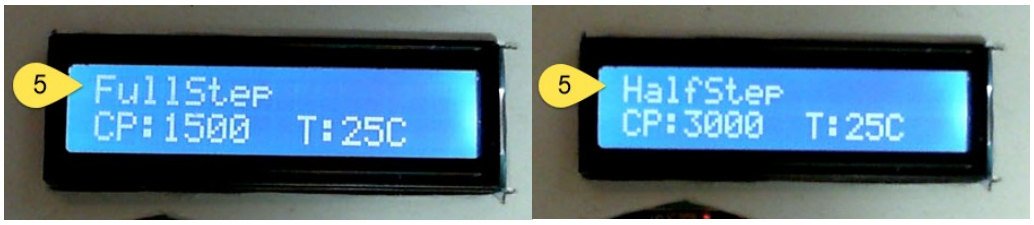
The Standalong PNP-Focus controller with the optional LCD1602 display shield provides a 16 character, 2 line visual status display. The following are display keywords and their meanings.



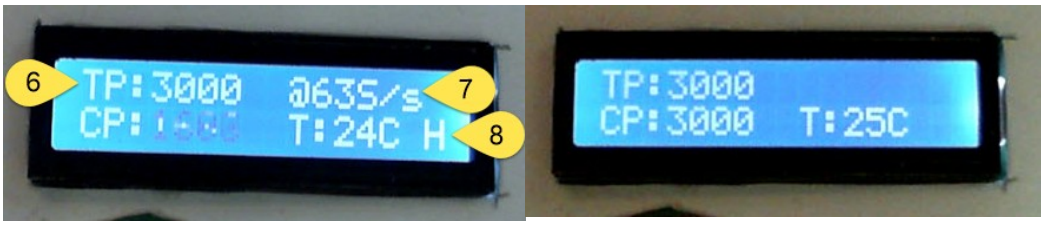
1. **PnPFocus** – appears on top line at left edge to identifies the unit name as PnPFocus on powerup.
2. **S/R3600 -** Appears on top row on right side only during the first Power- up. T to identify the firmware is programmed with the stepper rotation of 3600 steps/revolution. The value is set by the user when programming the unit and needs to match the focus stepper motor for the number of full steps per revolution. The display is provided verification of unit configuration. Useful you have multiple controllers for different stepper motor installations.
3. **CP:XXXX -** Appears on the bottom row at left edge and reports the **Current Position** value. The current position range is 0 – 65535 steps. You should measure your focuser travel from complete racked in to complete racked out position to determine you actual focuser movement limits. The current position value will change with each step of the motor given the motor. When the stepper mode is changed from full steps to half steps the current position will half and when changed to half step mode the current position will double the value at the change.

Increasing current position must always represent racking out of the focus position for PNP-Focus to operate correctly.

4. **T:20C or T:68F or T:N/A –** Appears on the bottom line on the right side to display the current temperature value in Celsius or Fahrenheit if the DS18B20 temperature sensor is installed. The value is the rounded to the nearest integer and is update either every 30 seconds for standalone, or every time the ASCOM driver queries the current temperature. If the DS18B20 sensor is not installed or there is problem communicating with the sensor, the display will indicate Not Available (N/A).



5. **FullStep or HalfStep** – appears on the top row when the down button is pressed and indicates the stepper move operation has changed to either full step mode or half step mode. When the stepper mode is changed from full steps to half steps the current position will double and when changed to half step mode the current position will half the value at the change so the current position will report the equivalent number of steps.



6. **TP:XXXX –** Appears on the top row at the left edge and reports the **Target Position** value. The Target position will be displayed any time the ASCOM drive issues a move to a specific position. The value will also display the current target position if temperature compensation is enabled and moved the focus position off of the last moved position. The Target Position value is used to manage all ASCOM driver position changes. Temperature compensation will change the TP value in the PNP-Focus to move the motor but will remain fixed the the last moved value in the ASCOM display. When a backlash value is entered in the user firmware settings, the target value remains fixed, but the current position will overshoot the target position, pause and then return back to the target position. A rack in position change will move direct to the target position. This ensures the target position is always approached from the same movement direction. When the ASCOM driver is disconnected the TP display will return to PNP-Focus to indicate system returned to full standalone operation.

7. **@XS/s** – Appears on the top row on the right side when the left or right buttons when a motor movement is requested or when a target move is commanded by the ASCOM driver. **X** displays the motor speed in Steps/second. The value displayed is always the current motor speed in

Steps per second. The speed dynamically change with left and Right button press holds for manual movements. The values also change with a ASCOM commanded move to manage a soft start and an approach to the target position. The motor speed value will disappear within 1 second of the motor stopping

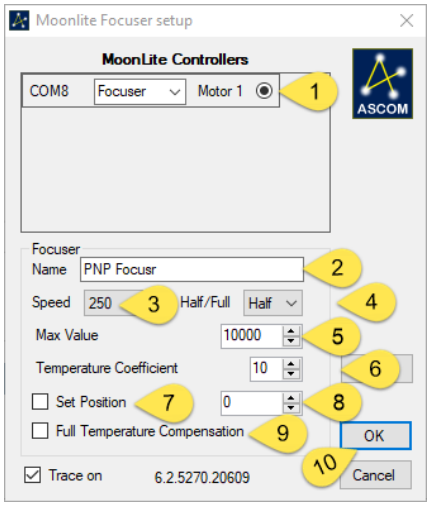
8. **H or F** – Appears on the bottom line at the last right character. When present, the H indicates the Ardumoto drive output stage is active in Half Step mode and power is applied to the motor. When present, the F indicates the Ardumoto drive output stage is active in Full Step mode and power is applied to the motor. When the commanded moves are compete, the drive will remain active for 3 seconds before being disabled.
9. **Stopped and Saved** – Appears on the top row when the select button is pressed to perform stop and save position function. The current position and the step mode values are written to non-volatile memory and will be restored on next power up.

Moonlite DRO ASCOM Controls

ASCOM DRO 6.0 Driver Download: **Moonlite DRO Setup.zip**
Reference Material: Hi Res Stepper Motor 2015.docx

The PNP-Focus is configured to accept and respond to commands issued by the Moonlite Telescopes Ltd. DRO ASCOM focus driver. The DRO driver is backward compatible to many of the moonlite focus controller product. As such there has been some issues connecting the PNP-Focus to the DRO driver. When the new version was released, the DRO driver would not recognize the PNP-Focus original firmware. The responses to serial interrogation by the driver appeared to be part of the problem. The DRO driver has some method of determining if the controller is a single or dual motor controller. Originally it appeared that the response to second controller requests and response to version was the reason for the issue. Testing was compared using an existing original moonlite mini focus controller. If the mini controller was ever connected to the computer the DRO focuser responded with single focus control configuration. If it was never connected to the testing computer, then the DRO responded with a dual controller configuration. The Moonlite Telescope single controller responded with version 13 and the documentation suggests the dual controller will respond with version 23. Even with trying these values in the firmware version, there was no consistent determination on how to invoke single controller configuration. The DRO interrogates all the serial ports and only displays ports which respond correctly upon initial setup interrogation.

DRO ASCOM Focus Driver Setup:



ComPort: After interrogating all the available ports only the focuser devices will appear in the list. You must select it to be used as a focuser and the click the button to make it active.

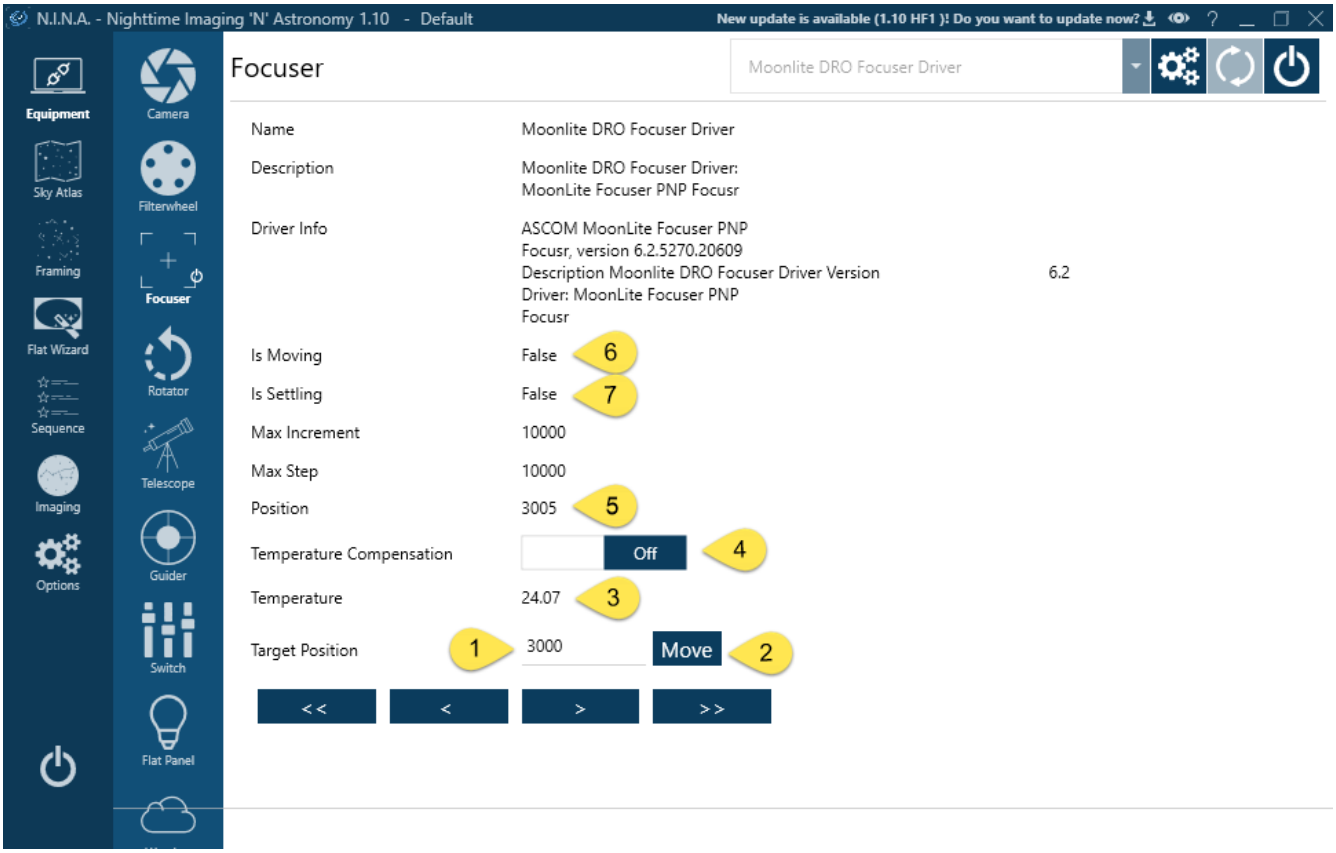
1. **Name:** Enter a name which defines this focus device. It might be primary telescope.
2. **Speed:** Select a speed value for the maximum speed for which PNP-Focus will drive. The values are 16, 32, 64, 125 and 250 steps / second. Because PNP-Focus uses a soft start and soft approach to target position, the value can be selected to a relatively high value. Often heavy focus tube loads can lead to focus tube slip if the speed is too high for extremely heavy loads. Determine the best value for equipment load and focuser configuration.
3. **Half / Full:** Sets the stepper operation mode when the focuser connection is established.
4. **Maximum Value:** Enter the value which is determined to be the maximum value which can be entered in the target position field. Usually the value will be the number of full steps or half steps which moves the focuser from fully racked in to racked out.
5. **Temperature Coefficient:** Enter a value which represents the amount of focus position change for the telescope and is expressed in steps per degrees. If the PNP-Focuser is set to display Celsius or Fahrenheit, the same unit is used for the coefficient. The value entered can be either positive or negative. Positive value results in increase position changes with increases in temperature changes. Record a number of focus changes with value set zero and reported temperature over the course of several nights to determine your telescope temperature coefficient.
6. **Set Position Enable:** The check box will enable the focus position to be set to the value in the position field when the PNP-Focus controller is connected. The set position will only apply once and then cleared once the command is passed to the controller.
7. **Position Field:** Enter the value for which the set position will assign the controller as the current position when making the connection to the PNP-Focus controller.
8. **Full Temperature Compensation:** When enabled all current positions are calculated as the corrected current position with the temperature coefficient. The value displayed in the ASCOM current position will not agree with the PNP-Focus position because the temperature coefficient has been applied to the value sent to the PNP-Focus controller. Also the Target position sent to the PNP-Focus will not match the values reported in the PNP-Focus display. Unlike the name suggests, a temperature change will report a target position change in the ASCOM target position but will **NOT** actually move the focus motor. The value represents the current equivalent focus position based on the temperature change. Using full temperature compensation will only move the focus to the calculated position and will not make any focus change for any further temperature change until the next position move. **HIGHLY RECOMMEND** to leave this option **DISABLED** unless you fully test the effect with your

system. It may not make sense and may NOT drive your telescope focus position the way you want.

If using the full temperature compensation you need to ensure your focus v-curves have been performed with full temperature compensation enabled during the focus curve recording. The focus will be moved to a position representative of the temperature change offset. Secondly the focus will not move after the target position move. The ASCOM current position value will now provide how much the focus position has shifted since the original target move. It can be used to indicate how much you are now out of focus, and provide some insight when to invoke the next focus routine. There are telescope operations where it makes sense to enable full temperature compensation. Test and confirm before use.

9. **OK:** When all the settings are set press OK to accept the settings and will be applied when the PNP-Focus controller is connected.

When the focus is connected, the ASCOM controlled program will have controls which allow you direct control of the focus motor and position. Every program has their own unique controls to implement the focus controls. The following example is how NINA employs the DRO focus driver controls. Other programs may be have different labels but the functions should be the same.



1. **Target Position:** Enter the step position you desire the focus motor to be driven. The target command will not be sent to the PNP-Focus until the move button is pressed. The value entered

is litterally the count of steps depending on the stepper mode in half or full steps. The ASCOM program may not indicate the current stepper operating mode.

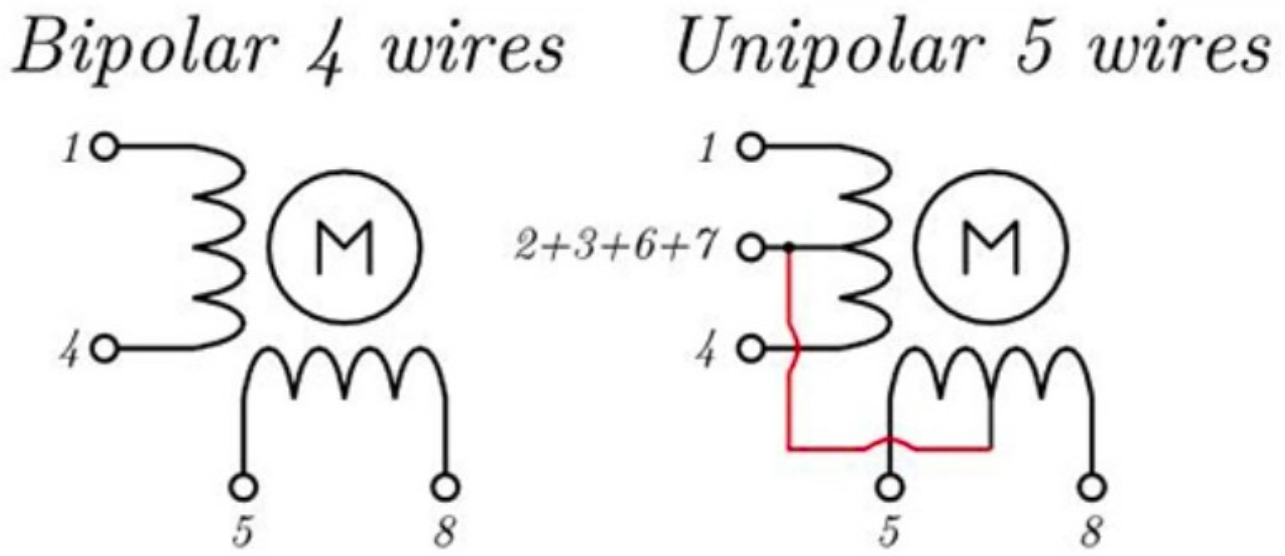
2. **Move Button:** When the button is pressed the value in the target position field is sent to the PNP-Focuser and executed as a move.
3. Temperature Value: The DRO focus driver interogates the PNP-Focus every 5 seconds and produces a running average for which temperature compensation calculations will be based. Remember this temperature may not fully represent a real air temperature. The temperature is based on where the temperature probe is located and the latency effects.
4. **Temperature compensation Enable ON-OFF control**. In this example temperature compensation is already enabled. With full temperature compensation disabled in this example, the current temperature compensation gets zeroed with every target position move. Any temperature change after the move will indicate a focus position change based on the temperature coefficient.
5. **Current Position:** Displays the current focus position. When a target position is set and move is executed the PNP-Focus will drive the focus to the target position and report the position. When temperature compensation is enabled and full temperature compensation is disabled, the current position is the value will change with temperature, the target position will not change however the current focus position will and the motor will move the focus tube to the current focus position. Thereby focus will move. This may be good or bad depending on your focus tube position bearings. If imaging it may however produce a position change if you have a bad focus tube. Temperature compensation can be a delight or a nemisis to Astroimaging depending on the equipment. Temperature compensations position changes occur very slowly and are not subject to backlash compensation actions.
6. **Is Moving:** Indicates focuser is moving based on a target position move command.
7. **Is Settling:** Is not provided by the DRO ASCOM focus driver. Therefore when PNP-Focus slows to approach the target, the Is Settling value is never reported.

Operation Principles

Choosing Stepper Motors

Motor Wire Configuration

Basically all stepper motors are DC armature motors with two coils which are 90 degrees apart from each other to provide the ability to step or cog the motor shaft by the step coil. The two most common motor configurations are Uni-Polar or Bi-Polar.



Bi-Polar motors only have two leads for each stepper coil. Voltage applied across the coil causes current to flow through the coil, inducing an electro-motive force which pulls the motor shaft to a position. In order to create the position step, the voltage must swap between the leads between alternating steps for the coil. For a 12V stepper in one step pin 1 is 12V and pin 4 is ground or 12V return. For the next step change for that coil the applied voltage must swap so Pin 4 is 12V and Pin 1 is ground or 12V return. Hence is why it is called Bi-Polar because each coil has two operating voltage polarizations.

Therefore the stepping sequence might look like

1	4	5	8
GND	12V	GND	12V
GND	12V	12V	GND
12V	GND	12V	GND
12V	GND	GND	12V

The Uni-Polar motor can come in a variety of wiring configurations. Uni-Polar motors only have one polarization where positive voltage is applied only to one lead per coil. If the motor is 5 wires then each coil has a centre tap and the two centre taps are tied together for the 5th wire lead. If it is a 6 wire, then each coil is centre tapped and each one coil centre tap is 5th lead and other centre tap is the 6th wire lead. Highly unusual but it could also come as a 8 wire stepper motor. In this situation each coil is actually made up of two coils of which each has a separate top and bottom lead. It then becomes very difficult to identify which lead is for which coil and which is the top and bottom of the coil to form the centre tap lead. Uni-polar motors are typically driven by a single voltage source and the voltage polarity never changes. Typically the center tap would have 12V or Vcc and then you just connect one of the leads to ground to cause current to flow through that one side of the coil to produce the step. With the Uni-Polar configuration only 1/2 of the coil is ever energized at any given time. The voltage application and stepping sequence for the Uni-Polar motor might look the the following:

6	1	4	5	8	7
12V	GND	Open	GND	Open	12V
12V	GND	Open	Open	GND	12V
12V	Open	GND	Open	GND	12V
12V	Open	GND	GND	Open	12V

Realize that current only flows from the center tap on pin 6 or 7 through the coil connected to ground (12V return path) producing the force to move the motor shaft. The open coil has voltage but no current flow hence no force to produce motor shaft rotation when open. With Uni-Polar stepper motors the voltage does not change to change direction of the motor shaft. Typically all that is one side can induce direction one way where the coil on the other side can induce the opposite direction or to pull onto the final stepped direction.

Converting a Uni-Polar to a Bi-Polar

Removing any voltage source from the the center tap of the each motor winding and moving it to the end of the winding then for all intensive purposes the motor becomes a bi-polar motor provided you put voltage reference on either side of the coil. Because current flows through more coil wingdings (2x) the electromotive force produced is double the force on the armature. In this way a Bi-Polar motor operation of a Uni-Polar motor will produce twice the amount of torque than operating in Uni-Polar mode because both coils are used to produce the Electro-motive force to move the motor shaft.

In the case of the moonlight focuser hi-resolution motor, leaving pin 5 open and not connecting it to 12V will produce 2x the torque for each step movement. This can be an important consideration especially thinking about moving heavy cameras, filter wheels and rotators connected to the focus tube. It would make sense to possibly wire a Uni-Polar motor in a Bi-Polar configuration. The ardumoto board applies a lead swap between the motor pins so + supply Voltage (positive lead) is on one side and return voltage path (negative lead) is on the other motor wire by changing the direction pin on the

board. The enable pin is always one when driving. Therefore the motor leads on the from the ardumoto board are always a voltage source and return on the two leads.

Stepper Motor Rotation Steps

The key when selecting the stepper motor for focus application is the number of steps per revolution. The steps per revolution is probably the most critical element in assembling your electric focusing system. You have to assess how does your telescope achieve critical focus and how much change is required to move you in and out of the critical focus. You need fine step resolution in your critical focus region. You ultimately should target at least 20 motor steps through this region. Start by assessing how much do you turn your focus know manually to go just in and out of focus. Also how are you installing your focus motor. Typically the stepper motor is installed on the main rotational shaft of your focuser say for a crawford focus tube. For something like an SCT, many of worked towards using a belt drive approach. You have to ultimately assess how much motor output shaft rotation is equivalent to your manual focus rotation in the critical focus region. For many refractor and newtonian telescopes you attach the motor direct to the main high speed focus control shaft. You do not attach it to a 10:1 knob shaft as those gears are not designed to take the rotation and torques that a motor will apply. Since the motor attaches to the primary shaft, you now have to remember how much your focus changes in one full rotation of the focus knob. The faster your telescope the more critical focus range is on the focus adjustment. Think about it for a moment. If you find you go in/out of focus with 1/4 turn of the fine focus knob. This means you are changing the main shaft by 1/4 of 1/10th rotation of the main focus shaft. Mathematically this is 0.025 of one full rotation. If you look at the shaft end on and remember a circle has 360 degrees rotation, this 0.025 rotation = 9 degrees of rotation. This notation is important, because stepper movement is always expressed in degrees of rotation per step or the number of steps for a full 360 degree rotation. Every full step will move the shaft by the documented rotation specification.

If you wanted 10 finer steps to define your focus range then you would need 9deg/ 10 or 0.9 degree/rotation stepper motor. Therefore when you go to select your stepper motor you want to look very closely at the stepper motor output rotation/step specification.

For example Moonlite Telescope high resolution focus motor output output shaft rotation is 0.1 degree per step, which is also expressed as 3600 steps per revolution. This means that for that 1/4 turn of fine turn knob this motor would use 90 steps to rack in/out of focus.

Hence it becomes very important to select a stepper motor with a high number of steps per revolution to provide fine step resolution for focus placement.

If you choose to use a belt drive mechanism, don’t forget to account for the reduction gain you can achieve by using different sized belt pulley wheels in your system design.

Full Step vs Half Step Motor Operation

To understand the differences between full step and half step, one has to visualize what must happen to make the motor shaft turn. The simplest terms is a stepper motor is a two phase motor with two electro-magnetic poles place 90 degrees from each other, relative to the motor shaft. Electric current through the a coil will produce a N-S magnetic orientation. The key to producing the N-S pole is the current flow direction from positive voltage to negative or zero voltage state. When current is flowing only through 1 coil (Phase A), the armature tries to align with the field of phase A. Energizing a single coil is shown as positions 2, 4, 6 and 8 in the adjacent diagram. When current flows through both coils (Phase A and B), there are now two magnetic fields and the rotor aligns between the two poles at 45 degrees. Energizing both coils at the same time is call a full step motor movement. When both coils are energized, there is twice the magnetic field interacting with the motor rotor and hence produces the strongest motor torque. Full torque, and full step state become position 1, 3, 5 and 7 in the adjacent diagram. The arrows at these positions are longer because the magnetic force is stronger than if a single coil was energized.

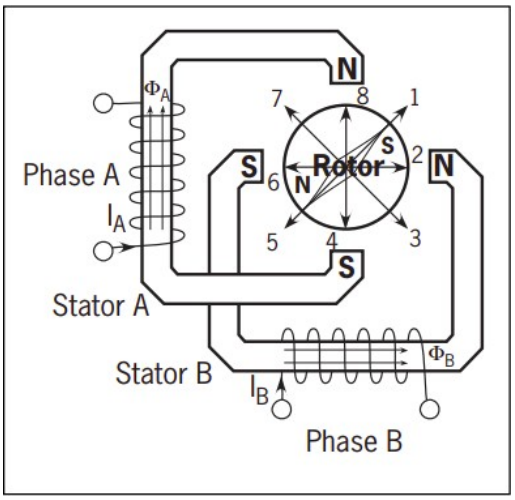
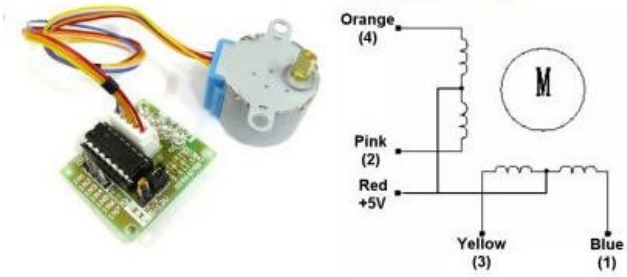


Figure 1: 2 Phase Stepper Motor

In order to produce motor rotation, the sequence of steps must energize the coils one direction or the other in the correct sequence to get the rotation. In the diagram full step sequence 1-3-5-7 would produce a Clockwise direction and 1-7-5-3 would produce a counter clockwise rotation. For Half steps it must be the sequential order 1-2-3-4-5-6-7-8 for clockwise and 1-8-7-6-5-4-3-2 for counter clockwise rotation. Most people fear and think connecting the wires to get the correct sequencing is hard to achieve. In actual fact, it's easy as long as always keep the wires for each coil together.

Looking at the 28BYJ-48 Stepper motor, the truth from the data sheets identifies the following sequence. The 28BYJ-48 has two versions of either 5V or 12V which is applied to red wire for a Uni-Polar configuration. The "–" in the truth table identifies the wire at 0 Volts to produce current flow through corresponding coil. It is important to realize stepping through step position 1 to 8 produce clockwise (CW) direction, and reverse order 8 to 1 produced counter clockwise (CCW) direction.



Half-Step Switching Sequence

Lead Wire Color	---> CW Direction (1-2 Phase)							
	1	2	3	4	5	6	7	8
4 Orange	-	-						-
3 Yellow		-	-	-				
2 Pink				-	-	-		
1 Blue						-	-	-

Figure 2: 28BJY-48 Stepper Motor

To de-mystify the wiring situation, the following is what happens when the 28BYJ-48 (12 volt version) is wired in 4 different ways to the PNP-Focus Ardumoto shield. Wires have to be connected in coil pairs where blue and yellow is one coil and orange and pink is second coil. Motor shield has motor A or B output for either coil. The table top is the Ardumoto terminal strip when viewed from the wires from left to right. All results are voltmeter measurements recorded on a bipolar connected 12V Hurst stepper motor. The LED states are on my standalone unit used two different colour LED being red (R) and yellow (Y). Te PNP-Focus Stick the Nano motor shield used same red colour so they were differentiated by Inner(I) as being closest to the terminal strip and Outer (O) being furthest away from terminal strip. The half step switching sequence is the 28BYJ-48 stepper motor state in figure 2 above where the dash represents the wire fired using a 0V state at the appropriate colour lead.

PNP Focus Display Step	Ardumoto Board (wire entry view)						Half Step Switching Seq #	Rota
	LED's		Terminal Strip (viewed L-R from wire)					
	B	A	3	4	2	1		
	Wire Colour ->		BLU	YEL	ORG	PNK		
1	R-O	OFF	12V	0V (-)	5V	5V	3	CCW Dir ...>
2	R-O	R-O	12V	0V (-)	0V (-)	12V	2	
3	OFF	R-O	5V	5V	0V (-)	12V	1	
4	Y-I	R-O	0V (-)	12V	0V (-)	12V	8	
5	Y-I	OFF	0V (-)	12V	5V	5V	7	
6	Y-I	Y-I	0V (-)	12V	12V	0V (-)	6	
7	OFF	Y-I	5V	5V	12V	0V (-)	5	
8	R-O	Y-I	12V	0V	12V	0V (-)	4	

Figure 3: Wiring #1

PNP Focus Display Step	Ardumoto Board (wire entry view)						Half Step Switching Seq #	Rotation
	LED's		Terminal Strip (viewed L-R from wire)					
	B	A	3	4	2	1		
	Wire Colour ->		YEL	BLU	ORG	PNK		
1	R-O	OFF	12V	0V (-)	5V	5V	7	CW Dir --->
2	R-O	R-O	12V	0V (-)	0V (-)	12V	8	
3	OFF	R-O	5V	5V	0V (-)	12V	1	
4	Y-I	R-O	0V (-)	12V	0V (-)	12V	2	
5	Y-I	OFF	0V (-)	12V	5V	5V	3	
6	Y-I	Y-I	0V (-)	12V	12V	0V (-)	4	
7	OFF	Y-I	5V	5V	12V	0V (-)	5	
8	R-O	Y-I	12V	0V	12V	0V (-)	6	

Figure 4: Wiring #2

PNP Focus Display Step	Ardumoto Board (wire entry view)						Half Step Switching Seq #	Rotation
	LED's		Terminal Strip (viewed L-R from wire)					
	B	A	3	4	2	1		
	Wire Colour ->		ORG	PNK	YEL	BLU		
1	R-O	OFF	12V	0V (-)	5V	5V	5	CCW Dir --->
2	R-O	R-O	12V	0V (-)	0V (-)	12V	4	
3	OFF	R-O	5V	5V	0V (-)	12V	3	
4	Y-I	R-O	0V (-)	12V	0V (-)	12V	2	
5	Y-I	OFF	0V (-)	12V	5V	5V	1	
6	Y-I	Y-I	0V (-)	12V	12V	0V (-)	8	
7	OFF	Y-I	5V	5V	12V	0V (-)	7	
8	R-O	Y-I	12V	0V	12V	0V (-)	6	

Figure 5: Wiring #3

PNP Focus Display Step	Ardumoto Board (wire entry view)						Half Step Switching Seq #	Rotation
	LED's		Terminal Strip (viewed L-R from wire)					
	B	A	3	4	2	1		
	Wire Colour ->		PNK	ORG	YEL	BLU		
1	R-O	OFF	12V	0V (-)	5V	5V	1	CW Dir --->
2	R-O	R-O	12V	0V (-)	0V (-)	12V	2	
3	OFF	R-O	5V	5V	0V (-)	12V	3	
4	Y-I	R-O	0V (-)	12V	0V (-)	12V	4	
5	Y-I	OFF	0V (-)	12V	5V	5V	5	
6	Y-I	Y-I	0V (-)	12V	12V	0V (-)	6	
7	OFF	Y-I	5V	5V	12V	0V (-)	7	
8	R-O	Y-I	12V	0V	12V	0V (-)	8	

Figure 6: Wiring #4

The test was the same for each sequence. Increment PNP-Focus from count #1 to cycles the output stages through the same switching sequence. Lights on each side of the terminal strip an indicate which direction the current flow is by either a Red or Yellow LED. The voltage state was measured for the sequence. The question becomes what occurs when wires are changed in different order?

In Wiring #1, step 1001, only the yellow wire is at 0V. Looking back at the truth table this corresponds to sequence step #3. The process is repeated for all the steps 1001 through 1008, looking at the 0V(-) correlation and recording the corresponding switching step number based on wire colour.

For wiring of blue - yellow – orange – pink sequence to the terminal, the step sequence is reduced sequentially in correct order and should produce counter clockwise (CCW) direction output on the motor shaft when counter positions are increased.

For the second test, wiring #2, the Yellow and Blue wires are swapped on the B3 and B4 output terminals to see if a valid step sequence is produced. The same sequence is performed and evaluated against the truth table. Again, a correct increasing step sequence is observed where steps complete a full proper loop. Therefore increasing display step values 1001-1008 should produce a clockwise (CW) motor shaft rotation.

For the third test, wiring #3, the pairs are swapped between output A and B, without lead swap in the coil pair to see if a valid step sequence is produced. The same sequence is performed and evaluated against the truth table. Again, like wiring #1, a correct increasing step sequence is observed where steps complete a full proper loop. Therefore increasing display step values 1001-1008 should produce a counter clockwise (CCW) motor shaft rotation.

For the fourth test, wiring #4, the orange and pink leads are swapped between output B3 and B4. The same sequence is performed and evaluated against the truth table. Again, like wiring #2, a correct increasing step sequence is observed where steps complete a full proper loop. Therefore increasing display step values 1001-1008 should produce a clockwise (CW) motor shaft rotation.

That confirms the wiring order mystery is not really a problem.

The second issue is Uni-Polar motor wiring. In all the tests above, if the red wire is connected to 12V, the lead that has 12V applied will not have any current flow to produce a N-S magnetic field. Only half of the coil, who's lead is connected to 0v(-) will produce current flow to produce the magnetic field. The current direction remains the same in all cases as the bi-polar stepper. The only difference is when the ardumoto shield has the output disabled, both output stages go to 5V instead of 12V. The current flows through each half of the coil and is balance so the magnetic field cancels each other out. So operating as Unipoled power to the 5th lead should work the same way but only half of the possible torque.

If the Uni-Polar motor red wire is disconnected and left open. First is to notice it is tied to both coils. Using standard resistance theory, the resistance of each coil is essentially the same. A standard voltage divider is formed and if 12V is place across either yellow/blue pairs or orange-pink pairs, the red wire would have half voltage, or 6V at the lead. In full step moves, both coils are energized and both would have the lead at a 6V potential. So there is no effect. When in half step the coil energized with 12V will produce 6V on red wire, however output was measured at 5V. The drive output is at a high impedence tri-state output, hence there is no current flow through the OFF coils and no magnetic field

Therefore, in summary.

- With any stepper motor, provided you ensure that both wires to a single coil wires are connected to the same motor output, A1-A2 or B3-B4, you will get a proper drive sequence. As soon as PNP-Focus drive the motor through 4 or 8 steps, it will find the correct rotational sequence to drive the motor.
- If using a Uni-Polar stepper motor, it doesn't matter if the 5th lead is tied to voltage or not as it will still produce a correct stepping sequence.
- If a Uni-Polar stepper motor had the 5th lead tied to Vin voltage, the motor torque output will be half of what a Bi-Polar drive would be.
- If a Uni-Polar stepper motor has the 5th lead disconnected, yes there will be a voltage produced on the lead and needs to protect it from contacting other metalic surfaces or PNP-electrical circuits. However it will function like a bi-polar motor and provide about twice the torque of the Uni-Polar motor operation. This mode becomes the recommended way to drive a Uni-polar stepper motor with the PNP-Focus unit.

All sounds good, but there is a catch! Half step drive on a stepper motor may not always have consistent step responses for each half step command. At various steps, only one stepper coil is firing, therefore there is reduced torque on the output shaft. Before considering using half step operating mode ensure that you repeatedly test your whole focus system with all equipment loaded on the focus tube to ensure the focus motor can move the load. Command the position repeatedly out and in and then physically measure if the focus tube is at the same position after a minimum of at least 10 cycles. If your load exceeds the stepper motor torque you will start to loose step positions even though the count is correct because the motor actually stalls. There is a higher probability for the stepper motor to stall with your focus tube load, in which you have inconsistent focus position changes when trying to use half step operations.

Recommendation is to select a correct motor with a higher number of steps, and run at full step movements; rather than use a lower resolution stepper motor and try to run with half step movements to provide the focus resolution needed to focus your telescope. The lower step motor will no doubt have lower torque and then asking it to use half steps, then uses even less torque for each step request. It becomes a double problem when considering designing and using half stepping operation with the stepper motor.

Backlash Compensation Considerations

With many geared stepper motors you may experience backlash situations. Backlash is simply the state that if the motor rotation changes, how much does the motor shaft have to move before it registers a change direction in the output shaft rotation. Very simple if moving from racking out how much does the motor need to turn before the focus tube start to move inward. Every system is different. Geared

focus tubes can have backlash, geared stepper motors can have backlash. The easiest way to deal with backlash is to ensure focus is always approach using the same focus movement direction. The predominant telescope design has the focus tube pointing down to the ground when the telescope is pointed up at the stars. In this manner, it is always preferred to move the focus tube up to achieve focus. The effect of gravity forces all the gears to be loaded the same way when focus is achieved. When the focuser is moved outward, the focus position is required to move beyond the target position and the reverse direction and rack inward to the finished focus position. In this manner backlash is compensated. If you are manually controlling the system using standalone system, you can use the focus drive buttons to manually compensate for any backlash issues. Computer control however requires a programmed compensation approach to deal with backlash.

The Moonlite Telescope DRO focus driver does not have a value or field to enter to provide backlash compensation. The Arduino code however has the backlash compensation value. The PNP-Focus is designed around the concept that an increase in position count is moving the focuser towards the ground and further away from the telescope objective. When the ASCOM driver is commanded to move to a focus target position is greater than the current position, it will apply the backlash value to it's first position move. Once the first position is reached the motor will reverse direction move towards the final focus position by reducing the focus position step count.

To illustrate if the current focus position is 1000 and backlash compensation value is 20; when the focus position is commanded to move to 1100, the focus motor will first drive out to 1120, stop and then move up to the final focus position of 1100. If the focus position were to move to 950, the focus would drive direct to position 950, without any overshoot. In this manner all target position have the focus moving upward to the final focus position.

Some telescopes however have focus tubes on top, where focus movements inward or smaller focus position number move in towards the ground. Example Newtonian telescopes with the focus tube on top. If you have this situation, the easy fix is to swap the stepper motor leads such that increases in focus position, actually move the focus tube inwards rather than outwards. Remember, backlash compensation will only apply to position where the target position is larger than the current position.

Therefore if you configure the motor direction where increasing the number moves the focus tube inward, then start your fully racked in position at your maximum movement count. For example, say your maximum movement position is 10000, start the racked in position at 10000. Then racking the focus tube up and out of the telescope would be some number less than 10000. This way a position move to 9000 would not have backlash compensation applied. However if the next position was 9100, backlash compensation would work properly. The position change from 9000 to 9100 is moving towards the ground and towards the pull of gravity.

Attachment to Focus Shaft

Attaching the stepper motor to your focus control shaft, has many different methods. If you are attaching to motor direct to the focus shaft, do not use a plain fixed coupler. The focus rods have a very small diameter. If the mount for the motor is not exactly perfect and the centre of the two shaft align, then no doubt the focus control shaft will be bent. Instead use some form of flexibile but rigid coupling between the motor and the focus control shaft. These coupling are very rigid in rotation however they are designed to flex in x and y direction as the rotation is performed. This ensures the focus position is accurate and that you do not have to exactly precise with the motor installation position. The couplings come in various designs and configuration. Many couplings have the same diameter on either side of the coupling, however many stepper motor output shaft is a different diameter than the focus control shaft diameter. Pay close attention to the hole diameter, when ordering the motor couplings. The most common coupling for stepper motors online is the 8mm to 5mm coupling where 5mm will attach to the stepper motor and the 8mm is designed to attach to a 3d printer shaft. The 8mm may not be the diameter of your focus control shaft.

AliExpress , Amazon , Ebay

Last item to consider is manual override. When you attach a geared stepper motor to a focus control shaft, you can not simply manually rotate the focus shaft know manually to move the focus. You must always use the motor drive to change the focus position. The other way to do it would be to use the Allan key and loosen the screws which connect the coupling to the focus control shaft. Moonlite Telescope found a clutch mechanism which can allow a no-tool disconnect of the focus motor to the focus control shaft. If you want to look at that approach, consider researching and using shaft to shaft clutches that can be purchased.

Shaft to Shaft Clutches: OnDrivesus.com

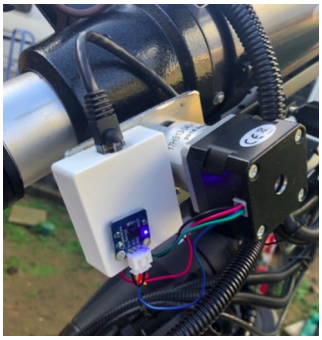
User Build Stepper Motors

The following stepper motors were listed as being used with user's build of the PNP focus controller.

Hurst ABS3008-002 Gear Reduction: 150:1, Output Step angle 0.1°, 3600 full steps/revolution, Coil Voltage: 12V, coils resistance: 53ohms, Coil amperage: 0.23A, Requires PNP-Focus power connection Vin=12V, used in RoboFocus Motor
Hurst Motors ,

Hurst - LSG35012F76P Gear Reduction: 75:1, Output Step angle 0.1°, 3600 full steps/revolution, Coil Voltage: 12V, coils resistance: 65ohms, Coil amperage: 0.18A, Requires PNP-Focus power connection Vin=12V, used in Moonlite Hi-Resolution focuser
Hurst Motors , Allied Electric

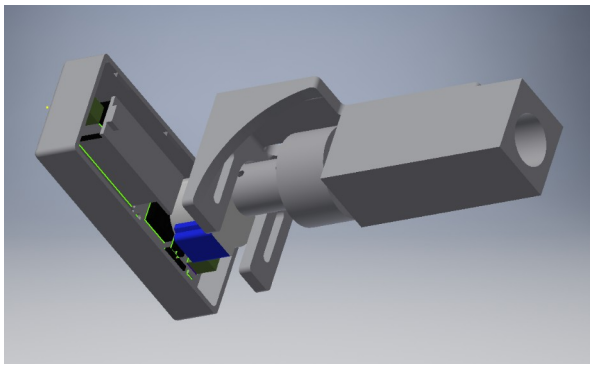
NEMA17 - 17HS13-0404S-PG27 Gear Reduction: 26.85:1, Output Step angle 0.067°, 5373 full steps/revolution, Coil Voltage: 12V, coil phase resistance: 30 ohms, Coil amperage: 0.4A, Requires PNP-Focus power connection Vin=12V
OMC-StepperOnline , ozrobotics.com Aliexpress

 installed by TomK

28BYJ-48 – 5V Gear Reduction: 64:1, Output Step angle 0.176°, 2038 full steps/revolution, Coil Voltage: 5V, coil phase resistance: 60 ohms, Coil phase amperage: 0.09A
Note: full step motor rotation =11.25°. Most tables list sequence for half steps. Half step output resolution = 0.088° Half step output resolution = 0.088° or 4076 steps/revolution which not what will be usable.

Note2: Confirmed was able to run 28BYJ-48 – 5V on standalone PNP-Focus running strictly on USB connection to an externally powered USB hub. Operates only in Full step mode without any special wiring or soldering to the ardumoto board pins. Lot's of torque and very quiet. The PNP-focus stick as stacked together does not provide 5V to Vin connection and will not drive the motor without

modification. However soldering a wire connection between 5V and Vin pins on the Nano motor shield, confirmed it will drive the motor off the USB connection. Now to just come up with housing to put the motor and PNP-focus all in one unit and attach directly to telescope. This becomes a completely self contained focuser. Caution would be always ensure it is run off an external powered hub. https://youtu.be/u_0s6xRi2YM



28BYJ-48 – 12V Gear Reduction: 64:1, Output Step angle 0.176°, 2038 full steps/revolution, Coil Voltage: 12V, coil phase resistance: 130 ohms, Coil phase amperage: 0.09A

Note: full step motor rotation =11.25°. Most tables list sequence for half steps. Half step output resolution = 0.088° or 4076 steps/revolution which not what will be usable.

Closing Comments

Thank you to Anat for introducing the PNP-Focus project. Thank you for all who have shared their success and progress with their project build. Pictures of users ideas and project builds, greatly help in information consolidation for this document. As time moves forward many of the details and items in this document will become stale and outdated. I can envision using the Leonardo and two Ardumoto shields to build a headless focus and rotator controller as a possible future adaptation. It would have to be headless because the LCD1602 shield used too many pins that would be needed to drive a second Ardumoto shield.

I hope this document will encourage you to take on your own PNP-focus controller project and that you will find success with your PNP-Focus controller.

Sincerely,
Todd Benko

Technical Reference

L298P 2A Dual Channel DC Stepper Motor Driver Shield Module For Arduino Nano 3.0

The NANO Motor Shield is based on the L298P, which is a dual full-bridge driver designed to drive inductive loads such as relays, solenoids, DC and stepping motors. It lets you drive two DC motors with your Arduino NANO board, controlling the speed and direction of each one independently. You can also measure the motor current absorption of each motor, among other features.

•You can find in the Getting Started section all the information you need to configure your board, use the Arduino Software (IDE), and start tinker with coding and electronics.

Technical specs

Operating Voltage	5V to 12V (See specification caution below)
Motor controller	L298P, Drives 2 DC motors or 1 stepper motor
Max current	2A per channel or 4A max (with external power supply)
Current sensing	1.65V/A
Free running stop and brake function	

Power

The NANO Motor Shield must be powered only by an external power supply. Because the L298 IC mounted on the shield has two separate power connections, one for the logic and one for the motor supply driver. The required motor current often exceeds the maximum USB current rating.
External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the Arduino's board power jack on which the motor shield is mounted or by connecting the wires that lead the power supply to the Vin and GND screw terminals, taking care to respect the polarities.
To avoid possible damage to the Arduino board on which the shield is mounted, we reccomend using an external power supply that provides a voltage between 7 and 12V. If your motor require more than 9V we recommend that you separate the power lines of the shield and the Arduino board on which the shield is mounted. This is possible by cutting the "Vin Connect" jumper placed on the back side of the shield. The absolute limit for the Vin at the screw terminals is 18V.
The shield can supply 2 amperes per channel, for a total of 4 amperes maximum.
(See caution specification below about power considerations)

Input and Output
This shield has two separate channels, called A and B, that each use 4 of the Arduino pins to drive or sense the motor. In total there are 8 pins in use on this shield. You can use each channel separately to drive two DC motors or combine them to drive one bipolar stepper motor. The shield's pins, divided by channel are shown in the table below:

Function	pins per Ch. A	pins per Ch. B
Direction	D12	D13
PWM	D3	D11
Brake	D9	D8
Current Sensing	A0	A1

If you don't need the Brake and the Current Sensing and you also need more pins for your application you can disable this features by cutting the respective jumpers on the back side of the shield.
Motors Connection

Brushed DC motor. You can drive two Brushed DC motors by connecting the two wires of each one in the (+) and (-) screw terminals for each channel A and B. In this way you can control its direction by setting HIGH or LOW the DIR A and DIR B pins, you can control the speed by varying the PWM A and PWM B duty cycle values. The Brake A and Brake B pins, if set HIGH, will effectively brake the DC motors rather than let them slow down by cutting the power. You can measure the current going through the DC motor by reading the SNS0 and SNS1 pins. On each channel will be a voltage proportional to the measured current, which can be read as a normal analog input, through the function analogRead() on the analog input A0 and A1. For your convenience it is calibrated to be 3.3V when the channel is delivering its maximum possible current, that is 2A.

Specification Caution
These reference notes appeared to be copied from many of the Ardumoto documentation. The silkcreen on the back of the board clearly states DC7-12V but yet documentation talks about Vin max of 18v.

A little more reverse engineering on the Nano motor shield board:
The Vin connection at the terminal strip passes through a reverse protection diode and then 2 filtering capacitors. The electrolytic capacitor is rated for 25v. The trace then jumps through the board and runs parallel A2 pin all the way to the Vin pin before making a trace connection to Vin. There is no Vin Connect jumper to be able to cut a trace to separate V-in from the Arduino micro controller as the documentation discribes.

looking at the chips:
LM358 (Closest to the electrolytic capacitor) datasheet specifies operating voltage 3-32V. OK
CD4077BM datasheet specifies DC supply range -0.5 to 20V OK
L298P datasheet specifies Vs input supply 50v Max OK.

Appears that all the chips on the motor control can handle voltages up to 20v. Therefore a 18V Vin would be a safe bet. So the problem clearly is the Vin connection to the Arduino Micro board

My gut feeling is 5V voltage regulator on the Arduino Micro probably can not handle voltages

in the 14v range. Since there is no simple disconnect of Vin on the micro board they labelled the V-in range 7-12V as recommended in arduino documentation.

Anyone see a problem developing here? People might be running their mount of a battery system with a charger applied to the battery. Typical battery charger will set voltage at 14.2V. So far I haven't seen this to be a problem with the standalone version. I have run my mount from a portable jumper pack and had it plugged into AC power and charging while it is running my whole mount and astro-photo rig. It didn't create any problem for the Stand alone version.

The Arduino Micro doesn't have a Vin barrel connection, and since we are using it strictly as computer connection, we need to seriously look at the Vin connection. It serves no purpose to have Vin connection from the Nano motor shield to the Micro. The Micro can be powered up and driven totally by the USB port. Therefore with the PNP-focus Stick, do not install a connector pin on the Arduino Micro Vin connection.

To drive the 5V version of the 28BYJ-48 stepper with the PNP-stick, we can simply solder a jumper between 5V and Vin on the back of the nano motor shield. So far I haven't had any problem driving the 28BYJ-48 from USB provided it is connected to an external powered USB hub.