Nema Stepper Motor 23 With Tb6600 Driver With Arduino Due

by Fernando Koyanagi

Today, we are going to talk about the Step Motor again. We will use a Nema 23 that will be controlled by a TB6600 Driver and an Arduino Due. It is possible to assemble powerful machines with this trio, and still keep costs low.

This Nema Step Engine has several versions. The 17 and 34, for example, are more expensive, and the 23, which we use today, is relatively inexpensive. But it has enough strength. The Nema 23 itself has several versions that reach 30 kgf.cm. Our example reaches on the tip of a spindle, that engine can push between 100 and 200 kg forward. In the video, we have the Step Engine spinning on account of a program that is already running. It’s next to the Arduino Due that is connected by four wires, one of them, referential, and the rest are signal. We have the Driver and a 24-volt by 10A power supply. I then put a Step Down Module in order to adjust the voltage on the Driver. A display then shows the voltage and current. This assembly is the first idea for a project that I want to set up: a Step Motor Laboratory.

15 kgf.cm.

So let's get the Nema 23 with the TB6600 Drive, which reaches its peak near 5A, which is dependent on its version. We'll soon make our engine work with the Arduino Due.

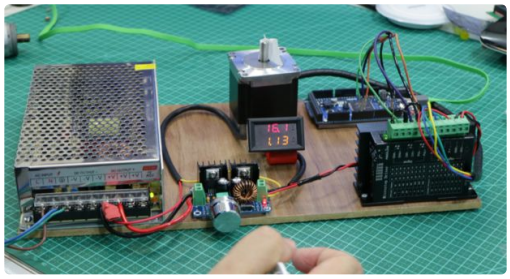
I see few people talk about the Arduino Due, but I like it a lot because it has a microcontroller with ARM Cortex-M3 core as the board's brain. This processor is far more powerful than the Arduino Uno. Cortex-M3, for example, gives the designer the ability to do more elaborate and sophisticated things.

Our assembly today, therefore, consists of the Arduino Due connected to the TB6600 Driver playing the Nema 23 Step Motor of 15 kgf.cm. If you put this

One feature of this TB6600 Driver is that it has a minimum operating voltage, which was 9 volts in the case of the one I used. The maximum I put with this Step Down is 19 volts, because this source is 24 volts, and it loses about 5 volts at this stage of regular voltage.

In this case, the motor is running at 1/32 micropasses. Just so you have an idea, a 3D printer works at 1/16 micropasses. So our Nema here is running very accurately and with a lot of force. Thus, Step Down controls the voltage and current, according to your preference. When the engine stops, a LED is lit for warning and protection of the Driver.

https://youtu.be/oqCNxC3LVsg



Step 1: Arduino Due

Talking about the Arduino Due, I need to point out that it is a Cortex-M3, and it features ARM processor architecture. This is open and widely used in mobile phones in general. I like this type because it is very optimized; it spends little energy and gives you processing power. It's also important to know that this Arduino is 3v3, different from Uno and Mega, the latter with its shield very similar to Due.

The Arduino Due has an input voltage of 6 to 20 volts because it has a voltage regulator. It has 54 IOs, which are input and output pins, 6 of which provide PWM. The A / D converter is 12 bits, and the output is analog, which means that I can generate a waveform, that is, I can play an MP3 or process a digital signal.

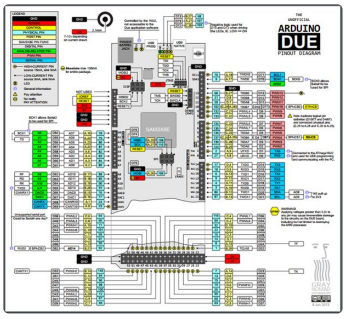
The consumption is high, about 800 mA. The Flash memory is 512 KB, and the RAM is 96 KB. It is important to remember that Arduino Due has no operating system. When you compile a C program for Arduino Due, you are compiling a self-contained program. It takes your program in C and generates a machine code that runs inside. It is different from you picking up a program and compiling on Raspberry Pi, where it will run on the Linux that is contained inside it.

Just to close out this presentation of Arduino Due, its clock speed is 84 MHz, and it has JTAG / SWD, with good debugging access.

One note: if you do not have the Arduino Due, you can use the Arduino Uno in this setup without any problems. I believe that the only thing that might need to be changed is the pinning.

Arduino Due

Arduino Mega



Step 2: How to Use the Stepper Engine and the Driver

For makers or those who just like to make their own designs, there are basically three situations involving the driver and stepper motor. The first is for when you already have a ready-made project, such as a Router machining a circuit board, in which you will not need to program. The second situation involves Motion Control: you put a camera that moves in time lapse, allowing you to control the cameras that are ideal for your purpose. The third possibility of creation involves industrial mechatronics: a stepper motor is transformed into a servomotor.

1. Mount a 3D Router

There is already firmware (grbl)

Control hardware already exists

Mechanical design already exists

Software integration is already in place

No control software

2. Motion Control

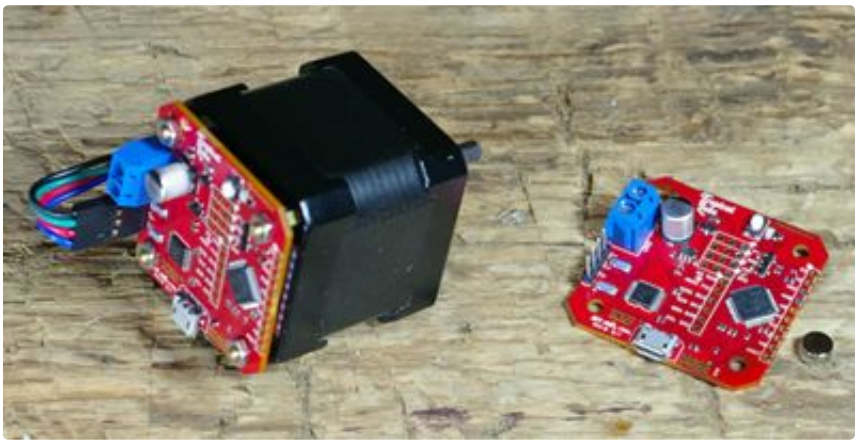
3. Industrial Mechatronics

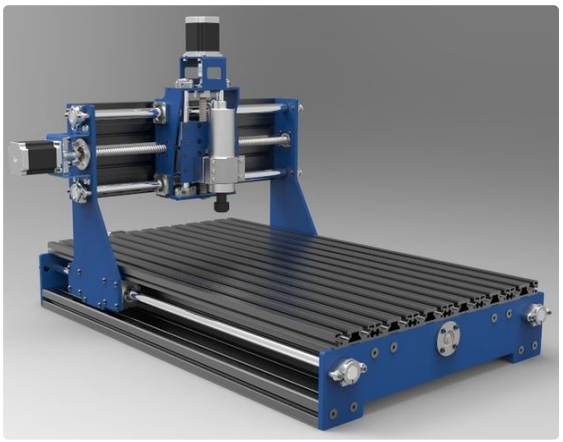
There is no firmware

Control board and firmwares are being exchanged:

There is no mechanical design

by Arduino and programming C

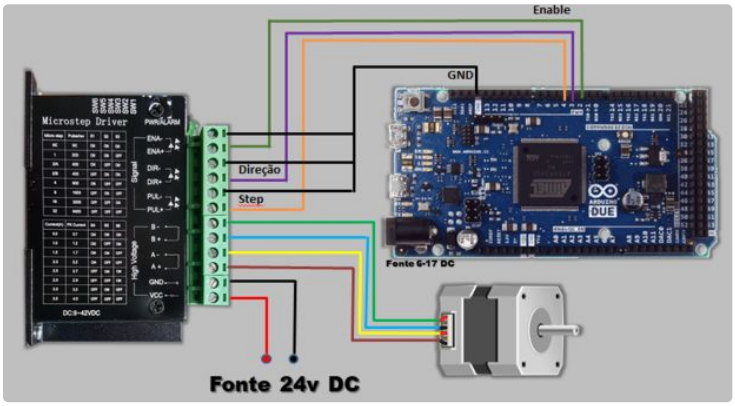




Step 3: Mounting With Arduino Due

Pay attention to the motor connection: AB, A +, A-, B +, B-, if it is with four wires, and the motor is bipolar.

Also, note the 3 connection pins of the Arduino Due with the Driver, including Steering and Step.



Step 4: Datasheet: Wotiom

Specifications:

Nema 23

- Torque: 15 kgf.cm

- Step Angle: 1.8º

- Number of phases: 2

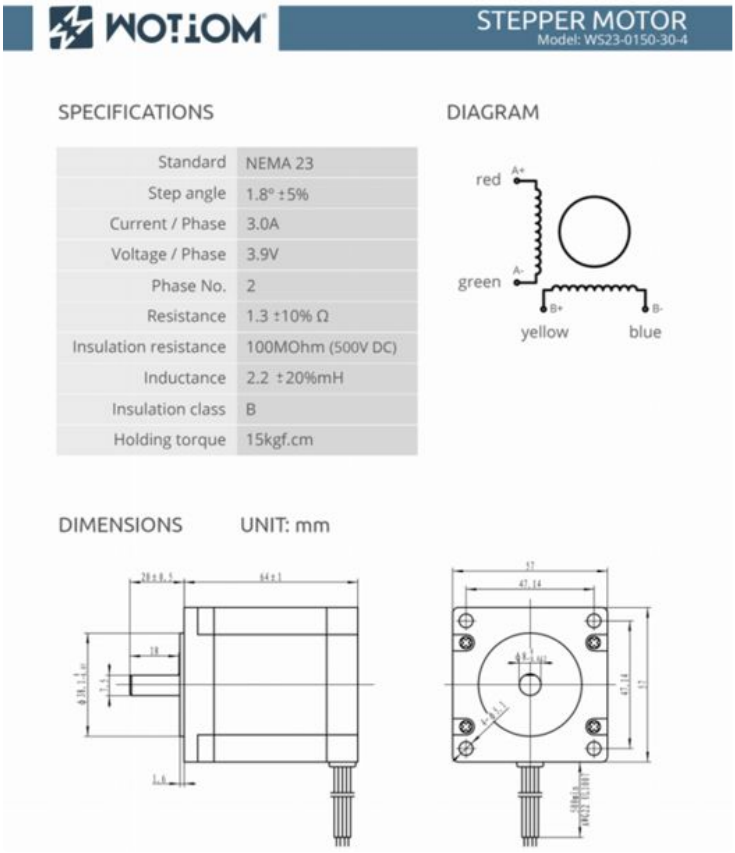
- Insulation class: B

- Insulation Resistance: 100M■

- Current by Phase: 3.0A

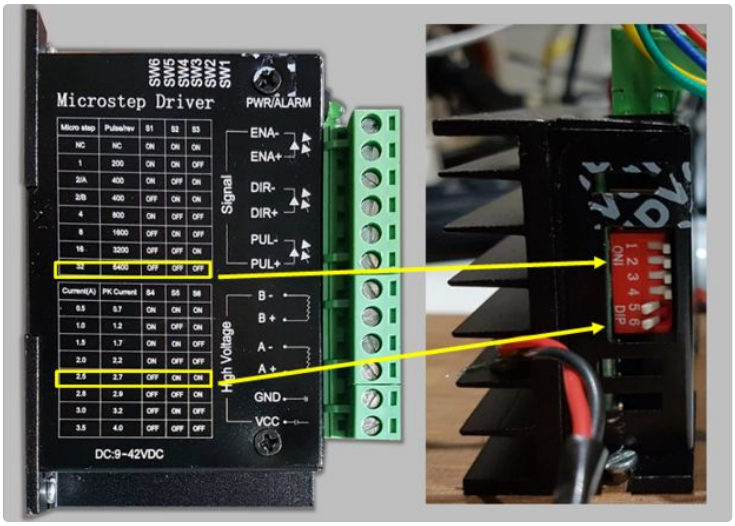
- Phase Resistance: 1.3■ ± 10%

- Phase Inductance: 2.2mH ± 20%



Step 5: Driver Configuration:

The configuration of this TB6600 Driver model is performed through the dip-switches. In this case, for example, my configuration is with 32 micropasses, 1/32, which should have the keys S1, S2, and S3 in the OFF / OFF / OFF positions. And then, for 2.5 amperes, dip-switches S4, S5, and S6 should be in the OFF / ON / ON position, respectively.



Step 6: Source Code

By setting the constants, we enable the motor, the direction, and the pulses. In the interval, we use 350 microseconds. We go to Setup, which in digitalWrite enables low, go to the Loop, which in the pulse will reverse the state of the variable, alternating between low / high, with a full wave in 700 microseconds. A complete explanation of the source code is shown in the video "Pt 3".

```
 const int ena = 2; //habilita o motor
 const int dir = 3; //determina a direção
const int pul = 4; //executa um passo
const int intervalo = 350;   //intervalo entre as
   // mudanças de estado do pulso
boolean pulso = LOW; //estado do pulso

void setup()
{
pinMode(ena, OUTPUT);
pinMode(dir, OUTPUT);
pinMode(pul, OUTPUT);
digitalWrite(ena, LOW); //habilita em low invertida
digitalWrite(dir, HIGH); // low CW / high CCW
digitalWrite(pul, HIGH); //borda de descida
}

void loop()
{
pulso = !pulso; //inverte o estado da variável
digitalWrite(pul, pulso); //atribui o novo estado à porta
delayMicroseconds(intervalo);
}
```