IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Implementation Of High Performance Micro Stepping Driver To Control Position And Speed Of Stepper Motor For Chemical Analyzer

P. Thimmaiah[1] , B. Ashraf Ahamed[2] and S. Khaja Hussain[3]

1 Assistant Professor, Department of Electronics, S.K, University, Anantapur, A.P. , India

2 Director, DR Biosciences, R & D Centre, Bangalore, Karnataka. India

3 Software Manager, Valeo India Pvt Ltd,. Chennai, India

Abstract: An automated analyzer is a medical laboratory instrument designed to measure various substances and other characteristics in several biological samples quickly, with minimal human assistance. These measured properties of blood and other fluids may be useful in diagnosing disease. These types of analyzers use micro-stepping motors for positioning the samples at equal intervals. Micro stepping is a method of controlling stepper motors, typically used to achieve higher resolution or smoother motion at low speeds. The movement of each step is precise and repeatable; therefore the motor's position can be controlled precisely without any feedback mechanism, as long as the motor is carefully sized to the application. This type of control eliminates the need for expensive sensing and feedback devices such as optical encoders. The position is known simply by keeping track of the input step pulses. It is one of the most versatile forms of positioning systems. The rotational motion of the stepping motor increases from one equilibrium position to the next. Therefore, the speed of a stepper motor is a function of the frequency at which the windings are energized. The drive design required to drive the stepper motor is taken into account with the choice.

Index Terms: Biomedical, stepper Motor, Micro driver, arduino, controlling, Micro steps, Instrumentation, Control System.

1. Introduction

The proposed system controls the motion and speed of the stepper motor by using an Arduino board with the help of a micro-stepping driver. With this technology, we can control both the direction and speed of a stepper motor. Manual control of the motor may cause errors, and it becomes tough for industrial applications and elderly or physically handicapped people to operate them. This system is enhanced to control the stepper motor through an Arduino board by programming the corresponding program. The proposed systems use an Arduino board, micro stepper driver, and MOSFET driver.

Industrial applications include high speed pick and place equipment and multi-axis CNC machines, often directly driving lead screws or ball screws. In the field of optics, they are frequently used in precision positioning equipment such as linear actuators, linear stages, rotation stages, goniometers, and mirror mounts. Other uses are in packaging machinery and positioning of valve pilot stages for fluid control systems. Commercially, stepper motors are used in floppy disk drives, flatbed scanners, computer printers, plotters, slot machines, image scanners, compact disc drives, and many more devices.

1.1 Stepper motor has its own characteristics

- It can be used for digital signal directly open-loop control, the entire system is simple and cheap.
- Directly receive digital signals, do not have a digital-analog conversion, easy to use.
- Displacement corresponds to the number of input pulse signals, not the long-term accumulation of step error, can be composed of a relatively simple structure but also has a certain precision of the open-loop control system and can also require higher accuracy when the composition of closed-loop control system
- Brushless, motor body parts less, high reliability
- Easy to start, stop, forward, reverse, and speed.
- Stop, can have a self-locking function.
- Large range of step angle selection and can be selected in dozens of angles to 180 degrees in a large range. In the case of small steps, usually at a lower speed under high torque operation, it cannot directly drive the reducer Load operation.
- Speed can be a wide range of smooth adjustments. At the same time, a controller to control several stepper motors can make them fully synchronized operation.
- It can't be directly used for ordinary AC power drives.

1.2 Stepper motor Driving and Control

Stepper motors require some external electrical components in order to run. These components typically include a power supply, logic sequencer, switching components, and a clock pulse source to determine the step rate. Many commercially available drives have integrated these components into a complete package. Some basic drive units have only the final power stage without the controller electronics to generate the proper step sequencing.

Common drives include Unipolar Drives, Bipolar Drives, L/R Drives, Chopper Drives, and Micro stepping Drives. Most commercially available stepper motor drivers take pulses as inputs. The amount of rotation of the stepping motor is proportional to the number of pulses given to the driver. The relationship of the stepping motor's rotation and input pulses are expressed as follows.

$$\theta = \theta s \times A$$

θ : Rotation angle of the motor output shaft [deg]
θs: Step angle [deg/step]
A : Pulse number [pulses]

The speed of the rotation is then proportional to the speed of the pulses. The relationship of the pulse speed (Hz) and motor speed (r/min) is expressed as follows:

$$N = \frac{\theta s}{360} \times f \times 60$$

N : Speed of the motor output shaft [r/min]
θs : Step angle [deg/step]
f : Pulse speed [Hz]
(Number of pulses input per second)

2. Proposed system

The TB6600 stepper motor driver is used to control larger two-phase bipolar stepper motors like NEMA 23 motors used in 3D printers, CNC machines, and robots. In this tutorial, I will describe the TB6600 motor driver hardware in detail and demonstrate how to control the driver with Arduino.

2.1 System Hardware Design

The TB6600 Stepper motor driver was originally built around the TB6600HG stepping motor IC made by Toshiba. However now a days many of these drivers have a TB67S109AFTG IC also made by Toshiba.

These chips are almost similar in performance and specifications but the TB6600HG is larger and has a higher peak current rating of up to 5A compared to the smaller TB67S109AFTG chip with a peak current rating of 4A. Also, the TB6600HG only supports up to 1/16 micro stepping while the TB67S109AFTG goes up to 1/32.

The driver has over-current, under-voltage shutdown, and overheating protection. Other specifications can differ slightly depending on the manufacturer and therefore you should always check the datasheet of your driver before use.

3. Connecting TB6600 Stepper motor driver to Arduino.

Figure 1 demonstrated the block diagram of Stepper Motor Driver with Arduino Board. Stepper Motor driver configured in a common cathode mode, all the negative sides of the control signal connected to the ground.

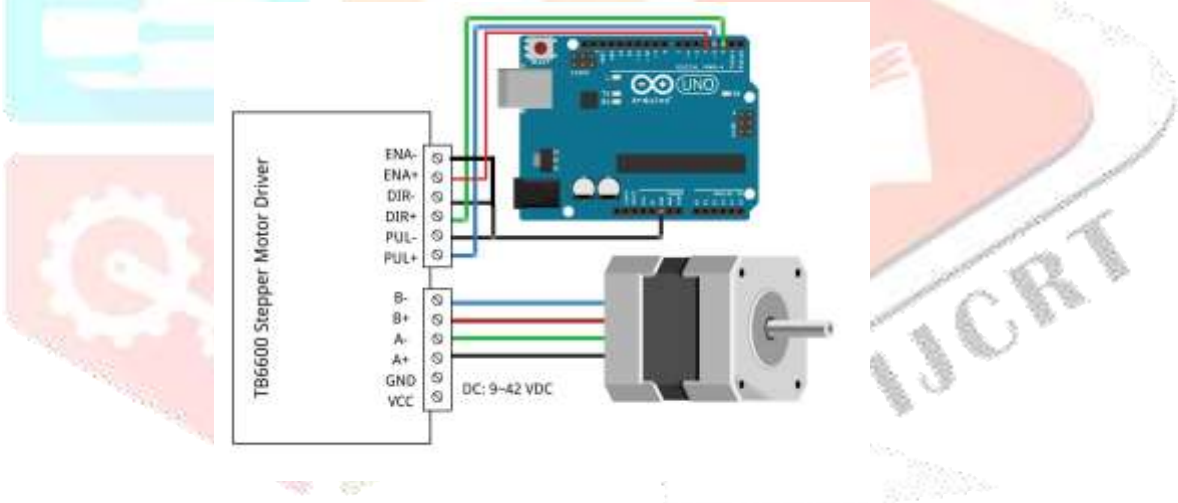

Figure 1: Block Diagram of the Stepper Motor Driver system

Table 1: TB6600 Stepper Motor Driver Connection

TB6600	Connection
VCC	9 – 42 VDC
GND	Power supply ground
ENA-	Arduino GND
ENA+	Pin 4 Arduino
DIR-	Arduino GND
DIR+	Pin 2 Arduino
PUL-	Arduino GND
PUL+	Pin 3 Arduino
A-, A+	Coil 1 stepper motor
B-, B+	Coil 2 stepper motor

Make sure that you do not connect stepper motors with a current rating of more than 3.5 A to the driver. A+,A- and B+,B- are the connections for the 4 wire bipolar stepper motor phases or coils. A pair of wires from one coil of the motor gets connected to A- and A+ and the other to B- and B+.

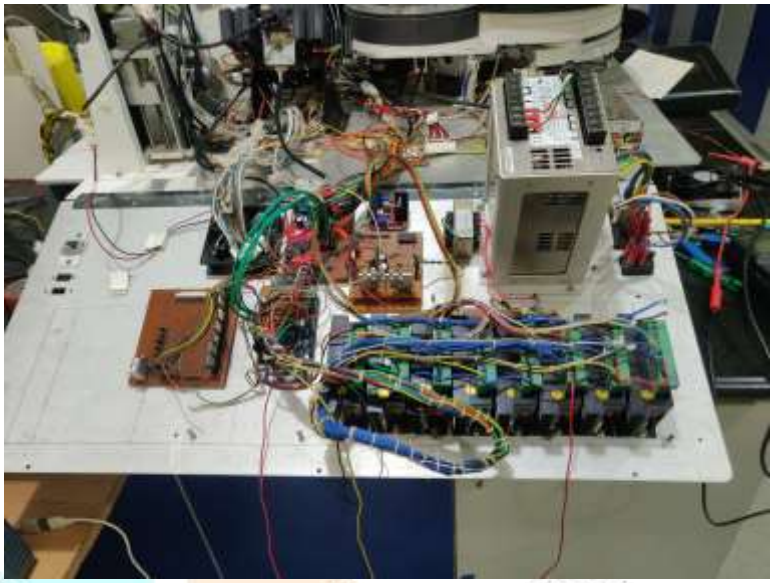

Figure 2: Stepper Motor Driver System Assembly

3.1 Adjusting Micro steps and Current

The TB6600 stepper motor driver has DIP switches used to set the microsteps and current depending on the specifications and application of the stepper motor being used. A table with microstep and current settings is printed on top of the driver case.

In full step mode, stepper motors normally make 200 steps per revolution, which gives a step size of 1.8°. The TB6600 driver supports micro stepping mode, which allows higher resolutions for stepper motors by allowing intermediate step locations through energizing the motor phases with intermediate current levels.

For example, when in ½ step mode, the stepper motor will make 400 microsteps per revolution and in ¼ step mode it will make 800 microsteps per revolution.

The TB6600 microstep settings can be changed by turning DIP switches S1, S2 and S3 on or off in specific order as shown in the table below. The settings below are for a 1/32 microstepping driver.

Table 2: TB6600 Micro Step Table

S1	S2	S3	Microstep resolution	Pulse/rev
ON	ON	ON	NC	NC
ON	ON	OFF	Full step	200
ON	OFF	ON	1/2 step	400
OFF	ON	ON	1/2 step	400
ON	OFF	OFF	1/4 step	800
OFF	ON	OFF	1/8 step	1600
OFF	OFF	ON	1/16 step	3200
OFF	OFF	OFF	1/32 step	6400

A smaller micro step setting will result in a smoother and quieter operation but will limit the top speed that you can achieve when controlling the stepper motor driver with a microcontroller. **Do not adjust the dip switches when the driver is powered.** Switches S4, S5 and S6 are used to adjust the current that goes to the motor when it is running. It is better to start with a current level of 1 A and incase the motor is missing steps or stalling; you can go on increasing the current level.

Table 3: Current Table

Current (A)	Peak current	S4	S5	S6
0.5	0.7	ON	ON	ON
1.0	1.2	ON	OFF	ON
1.5	1.7	ON	ON	OFF
2.0	2.2	ON	OFF	OFF
2.5	2.7	OFF	ON	ON
2.8	2.9	OFF	OFF	ON
3.0	3.2	OFF	ON	OFF
3.5	4.0	OFF	OFF	OFF

4. Code for controlling TB6600 Stepper motor driver with Arduino.

A microcontroller like Arduino can be used to control the speed, number of revolutions and direction of rotation of the stepper motor. The code below is an example of how this can be achieved.

```
const int stepPin = 3;
const int dirPin = 2;
const int enPin = 4;

void setup() {
  pinMode(stepPin,OUTPUT);
  pinMode(dirPin,OUTPUT);
  pinMode(enPin,OUTPUT);
  digitalWrite(enPin,LOW);
}

void loop() {
  digitalWrite(dirPin,HIGH); // Enables the motor to move in a particular direction
  for(int x = 0; x < 800; x++) {
    digitalWrite(stepPin,HIGH);
    delayMicroseconds(500);
    digitalWrite(stepPin,LOW);
    delayMicroseconds(500);
  }
  delay(1000); // One second delay
  digitalWrite(dirPin,LOW); //Changes the direction of rotation
  for(int x = 0; x < 800; x++) {
    digitalWrite(stepPin,HIGH);
    delayMicroseconds(500);
    digitalWrite(stepPin,LOW);
    delayMicroseconds(500);
  }
  delay(1000); // One second delay
}
```

4.1 Code description

First, the step (PUL+), direction (DIR+) and enable (ENA+) pins are declared with their corresponding Arduino pins as 3, 2, and 4 respectively.

In the setup() section, all the motor control pins are declared as digital OUTPUT and enable the driver by setting the enable pin LOW.

The loop() section determines the number of steps the motor will take. The four lines of code below will send a pulse to the step pin resulting in one microstep.

```
digitalWrite(stepPin, HIGH);
delayMicroseconds(500);
digitalWrite(stepPin, LOW);
delayMicroseconds(500);
```

4.2 Controlling the number of steps or revolutions

The for loop repeats the above lines of code a given number of times which represents the steps per revolution. For example, if the driver is set to ¼ step mode, then the code in the for loop has to be executed 800 times to get 1 revolution, that is,

```
for(int i = 0; i < 800; i++) {
   digitalWrite(stepPin,HIGH);
   delayMicroseconds(500);
   digitalWrite(stepPin,LOW);
   delayMicroseconds(500);
  }
```

If the motor is set to 1/8 step mode, then the above for loop would result in half a revolution.

4.3 Controlling motor rotation direction

The direction of rotation of the stepper motor is determined by setting the direction (DIR+) pin either HIGH or LOW, when the DIR pin is set HIGH, the motor will turn clockwise and when set LOW it turns counter clockwise.

4.4 Controlling speed of rotation of the motor

The speed of the stepper motor is determined by the frequency of the pulses we send to the STEP pin which is set using the **delayMicroseconds()** function. The shorter the delay, the higher the frequency and therefore the faster the motor runs.

4.5 Controlling TB6600 Driver using AccelStepper library

Mike McCauley has written the AccelStepper library that contains a number of functions that simplify the control of stepper motors with Arduino using TB6600 driver and many other stepper motor drivers.

This library can be installed directly from the Arduino IDE by going to **Tools > Manage Libraries…** to open the Library Manager where you can search for ‘AccelStepper’ and look for and install the latest version of the library by Mike McCauley.

Example code for controlling TB6600 Stepper motor driver with Arduino using AccelStepper library.

The code below is for moving the motor back and forth with a speed of 1000 steps/s and an acceleration of 500 steps/s^2. The driver is in ¼ step mode but you can change the mode and the speed and acceleration settings and observe what happens.

```
#include <AccelStepper.h>
#define dirPin 2
#define stepPin 3
#define enPin 4
#define motorInterfaceType 1

// Create a new instance of the AccelStepper class:
AccelStepper stepper = AccelStepper(motorInterfaceType, stepPin, dirPin);

void setup() {
stepper.setMaxSpeed(1000);
stepper.setAcceleration(500);
}
void loop() {
stepper.moveTo(8000);
stepper.runToPosition();
delay(1000);
// Move back to zero:
stepper.moveTo(0);
stepper.runToPosition();
delay(1000);
}
```

4.6 Code description

First, you have to include the AccelStepper library and then declare the TB6600 and Arduino connections and the motor interface type. When using a step and direction driver, the motor interface type must be set to 1.

Then you need to create a new instance of the AccelStepper class with the appropriate motor interface type and connections. You can even create multiple instances of the AccelStepper class with different names and pins which allows you to easily control two or more stepper motors at the same time.

In the setup() section, we use the **setMaxSpeed()** and **setAcceleration()** functions to set the maximum speed and acceleration or deceleration. In the loop section, the **moveTo()** function is used to set the target position in steps of the motor and the **runToPostion()** function moves the motor to the target position with the set acceleration or deceleration.

5. Result and Conclusion

This setup of hardware, stepper motor, and micro stepper driver (TB6600) is successfully interfaced with the Arduino board and the result was obtained as per application needs such as the stepper motor rotating in forward and backward direction and controlling the angle. This design of the stepper motor control system analysed the mechanism, working principle, the access to the stepper motor control system related scientific literature. This system followed the practical, simple, reliable, and low-cost principle, and designed a structure that is a very effective stepper motor control system. Through the drive circuit analysis and the physical control of the actual detection, this paper will have inference meaning for the stepper motor control system development.

References

[1] Stepper Motor, Wikipedia, http://en.wikipedia.org/wiki/Stepper_motor
[2] HSI Stepper Motor Theory, Haydon Kerk, https://www.haydonkerkpittman.com/learningzone/technicaldocuments/stepper-motor-theory#:
[3] An Introduction to Stepper Motors, Xinda Hu, https://wp.optics.arizona.edu/optomech/wp-content/uploads/sites/53/2016/10/Tutorial_Xinda-Hu.pdf
[4] TB6600 Stepper Motor Motor Driver with Arduino, https://mytectutor.com/tb6600-stepper-motor-driver-with-arduino/
[5] TB6600 Stepper Motor Driver : Pin Configuration, Interface with Arduino, Working & Its Applications, https://www.watelectronics.com/tb6600-stepper-motor-driver-module/
[6] Stepper Motors Basics: Types, Uses, and Working Principles, https://www.monolithicpower.com/en/stepper-motors-basics-types-uses
[7] PC Based Wireless Stepper Motor Control, Shanwaz Khan, http://www.diva-portal.org/smash/get/diva2:832112/FULLTEXT01.pdf
[8] Shailesh J. Parmar, Mital S. Zala, Design and Development of Stepper Motor Position Control using Arduino Mega 2560 [J], International Journal of Science Technology & Engineering, Volume 3, Issue 09, March 2017
[9] Swapnali, Ramesh Controlling Stepper Motor using Arduino Uno [J], nternational Research Journal of Engineering and Technology, Volume: 05 Issue: 04 | Apr-2018
[10] Ashwini S.Gavade, Sayali S. Bidawayi, Design and Implementation Interfacing Stepper Motor with Arduino [J], International Research Journal of Engineering and Technology, Volume: 05, Issue: 04 , Apr-2018
[11] Li Guohou. Stepper motor drive and control system design [J]. Coal Mine Machinery, 2008 (02)
[12] Liu Xinghui, Bi Guoling. Development of single-chip microcomputer control system of Stepper motor [J]. Journal of Liaoning University (Natural Science Edition), 2009 (03)
[13] Zhu Haijun, Zhang Shuocheng. Design and Application of Stepper motor Control System [J]. Nuclear Technology. 2005 (06)
[14] Ding Weixiong, Yang Dingan, Song Xiaoguang. Progressing principle of stepper motor and itsrealization based on single chip micro-computer [J]. Coal Mine Machinery, 2011 (03)
[15] Huo Yinghui, Chen Yuxiang. Microcomputer and single chip microcomputer control of stepper motor [J]. Application Research of Computers, 2005 (01)