Lab 4: Stepper Motor Control
Instructor: Dr. Carl Latino
Fall 2019

Goals

1. Understand the limitation of GPIO output current.
2. Learn to use Darlington transistor arrays to perform high-current driving with extremely low input current.
3. Understand the usage of full stepping and half stepping to control the speed and position of a stepper motor.
4. Gain experience of generating pulse waveforms to control a stepper motor.

Grading Rubrics (Total = 100 points)

1. **Pre-lab assignment:** 10 points.
2. **Attendance and Class Participation:** 10 points.
3. **Code Organization:** 10 points.
4. **First Objective:** 55 points.
5. **Second Objective:** 15 points.

Pre-Lab Assignment

1. Read the textbook Chapter 16 Stepper Motor.
2. Watch this video tutorial (8 minutes): How brushed DC motors are made and how they operate (Credit goes to *http://www.pcbheaven.com/*):
    a. https://youtu.be/RAc1RYilugI
3. Watch video tutorial: How the Stepper motors are made and how they operate (Credit goes to *http://www.pcbheaven.com/*):
    a. Part 1 (5 minutes): http://www.youtube.com/watch?v=MHdz3c6KLrg
    b. Part 2 (8 minutes): http://www.youtube.com/watch?v=t-3VnLadIbc
4. **Answer the pre-lab questions (10 points).**

Lab Objectives

1. **First Objective (55 points):**
    a. **Due date:**
        i. **For Mondays lab students:** March 25, 2019.
        ii. **For Wednesdays lab students:** March 27, 2019.

    b. Turn the stepper motor EXACTLY 360 degrees clockwise by using **full-stepping.**

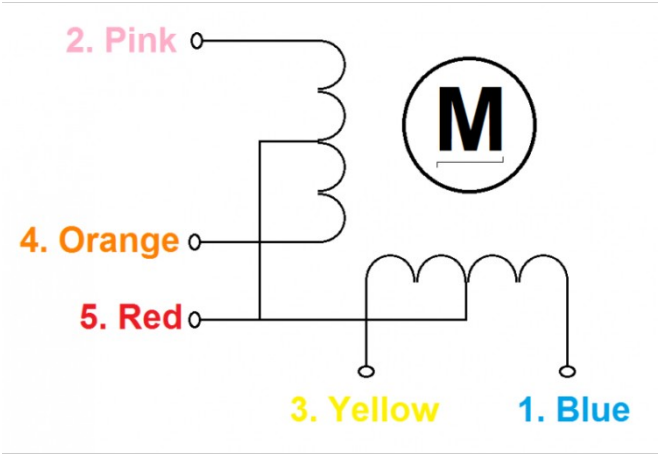
2. **Second Objective (15 points):** Choose ONE of the following options.
    a. **Due date:**
        i. **For Mondays lab students:** March 25, 2019.
        ii. **For Wednesdays lab students:** March 27, 2019.

    b. Re-implement the first objective using Assembly language.

    c. Use the joystick to control the motor in the following way:
        i. Make the motor move **clockwise using full-stepping** when the **right button** is pressed.
        ii. Make the motor move **counter-clockwise using full-stepping** when the **left button** is pressed.
        iii. Make the motor move **clockwise using half-stepping** when the **up button** is pressed.
        iv. Make the motor move **counter-clockwise using half-stepping** when the **down button** is pressed.

        Note: For each button press, the motor should only do ONE full rotation.

Stepper Motors



The motor has a ULN2003 Darlington Array.

Motor model	**28BYJ-48**	Number of phases	2
Rated voltage	5V DC	Geared reduction ratio	1/64
DC resistance per phase	50Ω±7%(25℃)	Pull in torque	>300gf.cm / 5VDC 100pp

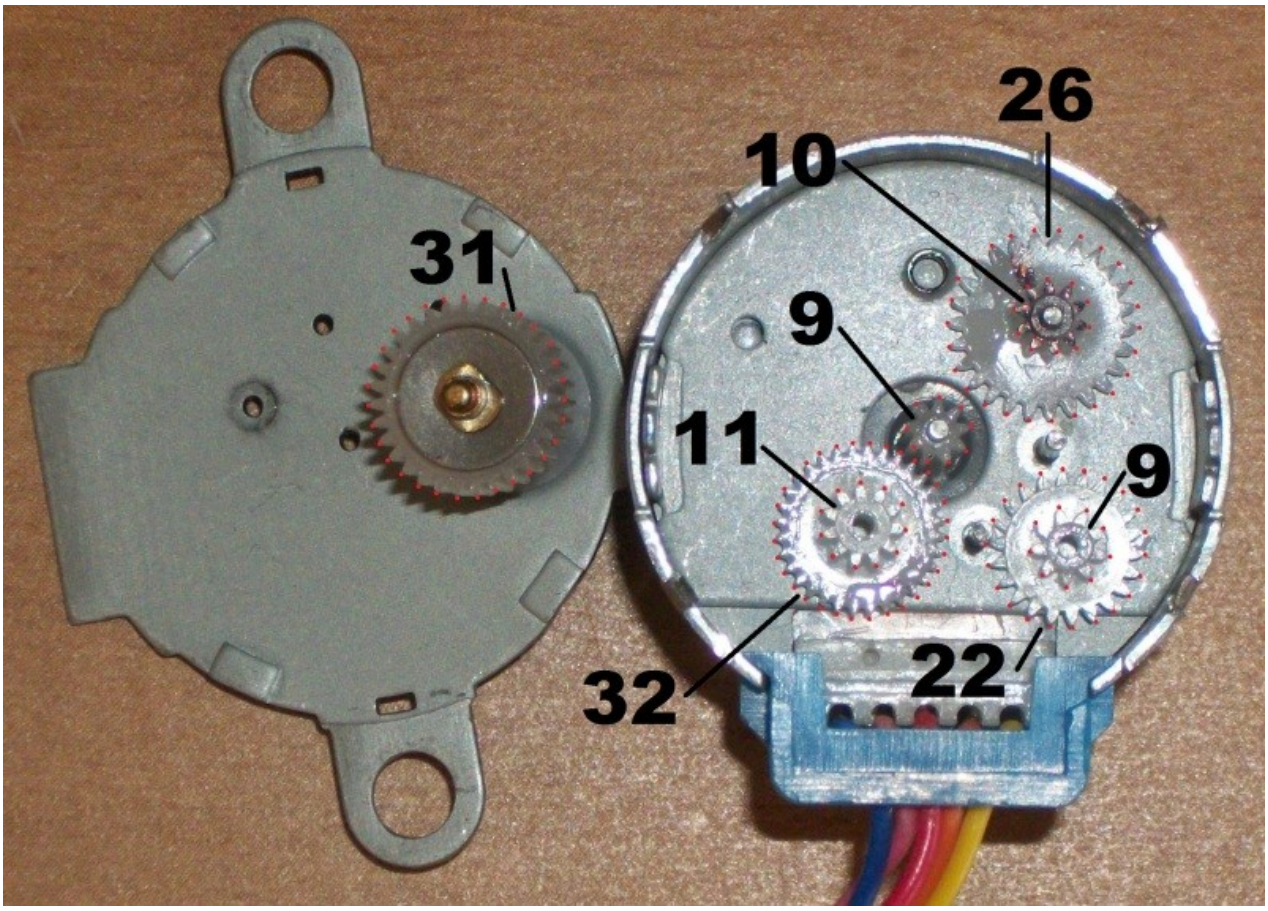


image from *forum.arduino.cc*

The gear ratio is:

$$\frac{32\times32\times26\times22}{11\times10\times9\times9}=63.68395$$

If the output shaft rotates 1 resolution (gear with 31 teeth in the figure), the internal shaft (gear with 9 teeth in the middle) must rotate approximately 64 resolutions.

Full-stepping

- Internal motor: 32 steps per revolution.
- Great reduction ratio: 1/63.68395, approximately 1/64.
- Thus, it takes $32\times64=2048$ steps per revolution for the output shaft.

Half-stepping

- Internal motor: 64 steps per revolution.
- Great reduction ratio: 1/63.68395 ≈ 1/64.
- Thus, it takes $64\times64=4096$ steps per revolution for the output shaft.

Lab 2: Lab Assignment

- A startup code in a zip-file (filename: *Lab 2 – Startup Code.zip*) is available on **D2L**. It contains the following files: **main.c**, **SysClock.c, SysClock.h,** and **stm32l476xx.h**.
    - **Download** and **extract** the startup code.
    - Create a new C Project using System Workbench for STM32 IDE.
    - Move the files **main.c** and **SysClock.c** to your project's **src** folder.
    - Move the files **SysClock.h** and **stm32l476xx.h** to your project's **inc** folder.
- In order to complete the lab objectives, you only need to write code in the **main.c** file.
    - For the first objective, you should complete two functions: ***GPIO_Motor_Init()*** and ***Clockwise_Full_Stepping()***.
        - **GPIO_Init():**
            - Set up the GPIO port B to be used to control the stepper motor.
        - ***Clockwise_Full_Stepping()***:
            - This function will effectively move the stepper motor.
            - You should complete this function by following the textbook's section 16.4.
    - For the second objective, you will have to figure out the code by yourself.
- **Academic Integrity Notice:**
    - Students are supposed to work individually! Copied code will incur in reduced grade!

Warning: Motor Overheating

The motor constantly draws electrical currents. The motor will be overheated if you leave the power on for an extended period. **Make sure to disconnect the power (Vcc) to the Darlington array if you are not debugging/testing it.**