23.Stepper_Motor

Introduction

Stepper motors, due to their unique design, can be controlled to a high degree of accuracy without any feedback mechanisms. The shaft of a stepper, mounted with a series of magnets, is controlled by a series of electromagnetic coils that are charged positively and negatively in a specific sequence, precisely moving it forward or backward in small "steps".

Hardware Required

- ✓ 1 * Raspberry Pi
- ✓ 1 * T-Extension Board
- ✓ 1 * Power Module (with 9V Power Adapter or 9V Battery and Buckle)
- ✓ 1 * Stepper Motor
- ✓ 1 * ULN2003 Driver Board
- ✓ 1 * 40-pin Cable
- ✓ Several Jumper Wires
- ✓ 1 * Breadboard

Principle

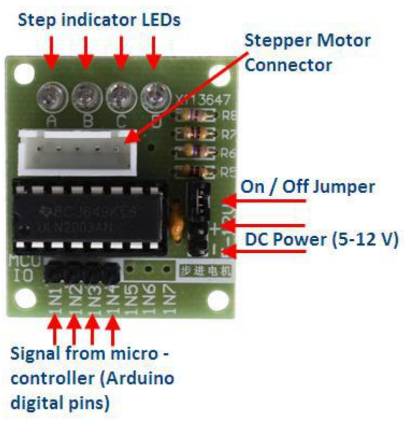
28BYJ-48 Stepper Motor

The 28BYJ-48 is a small, 5 volt geared stepping motors. These stepping motors are apparently widely used to control things like automated blinds, A/C units and are mass produced. stepper01 Due to the gear reduction ratio of *approximately* 64:1 it offers decent torque for its size at speeds of about 15 rotations per minute (RPM). With some software "trickery" to accelerate gradually and a higher voltage power source (I tested them with 12 volts DC) I was able to get about 25+ RPM. These little steppers can be purchased together with a small breakout board for the Arduino compatible ULN2003 stepper motor driver. Quite a bargain, compared to the price of a geared DC motor, a DC motor controller and a wheel encoder! The low cost and small size makes the 28BYJ-48 an ideal option for small robotic applications, and an excellent introduction to stepper motor control with Arduino. Here are the detailed specs of the 28BYJ-48 stepper motor.

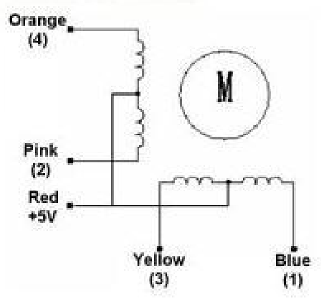
Stepper motor 28BYJ-48 Parameters

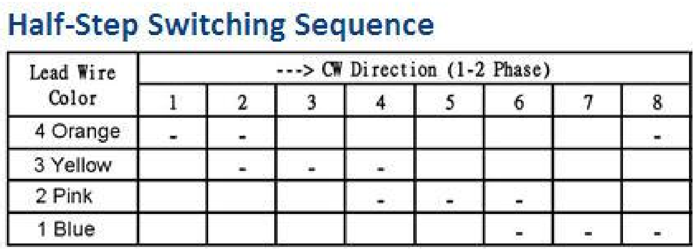
- ✓ Model:28BYJ-48
- ✓ Rated voltage:5VDC
- ✓ Number of Phase:4
- ✓ Speed Variation Ratio:1/64
- ✓ Stride Angle:5.625°/64
- ✓ Frequency:100Hz
- ✓ DC resistance:50Ω±7%(25°C)
- ✓ Idle In-traction Frequency:>600Hz
- ✓ Idle Out-traction Frequency:>1000Hz
- ✓ In-traction Torque >34.3mN.m(120Hz)
- ✓ Self-positioning Torque >34.3mN.m
- ✓ Friction torque:600-1200 gf.cm
- ✓ Pull in torque:300 gf.cm
- ✓ Insulated resistance >10MΩ(500V)
- ✓ Insulated electricity power:600VAC/1mA/1s
- ✓ Insulation grade:A
- ✓ Rise in Temperature <40K(120Hz)
- ✓ Noise<35dB(120Hz,No load,10cm)

ULN2003 Driver Board



The ULN2003 stepper motor driver board allows you to easily control the 28BYJ-48 stepper motor from a microcontroller, like the Arduino Uno. One side of the board side has a 5 wire socket where the cable from the stepper motor hooks up and 4 LEDs to indicate which coil is currently powered. The motor cable only goes in one way, which always helps. UNL2003 board On the side you have a motor on / off jumper (keep it on to enable power to the stepper). The two pins below the 4 resistors, is where you provide power to the stepper. Note that powering the stepper from the 5 V rail of the Arduino is not recommended. A separate 5-12 V 1 Amp power supply or battery pack should be used, as the motor may drain more current than the microcontroller can handle and could potentially damage it. In the middle of the board we have the ULN2003 chip. At the bottom are the 4 control inputs that should be connected to four Arduino digital pins.



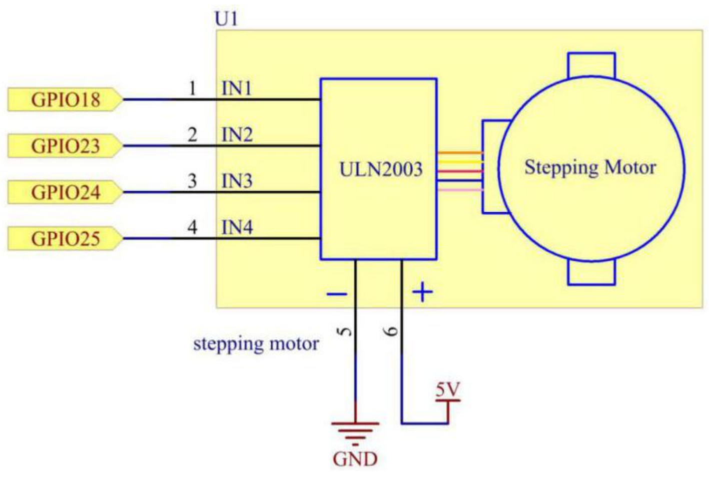


Half-Step Switching Sequence

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

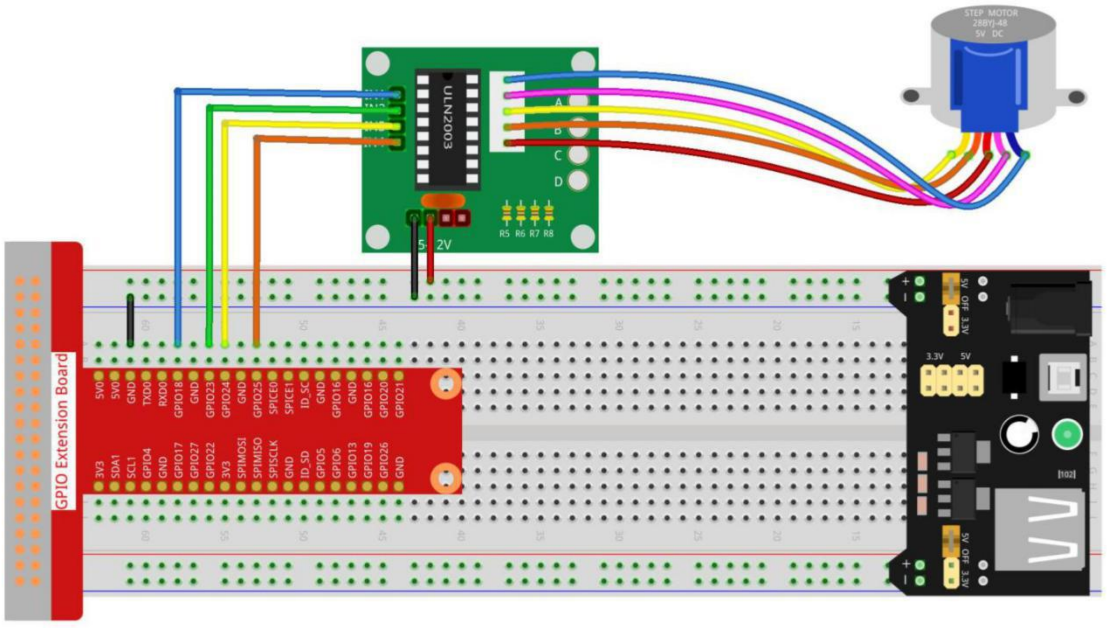
Schematic Diagram

T-Board Name	physical	wiringPi	BCM
GPIO18	Pin 12	1	18
GPIO23	Pin 16	4	23
GPIO24	Pin 18	5	24
GPIO25	Pin 22	6	25



Experimental Procedures

Step 1: Build the circuit.



For C Language Users

Step 2: Get into the folder of the code.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/C/23.Stepper_Motor
```

Step 3: Compile the code.

```
gcc 23.Stepper_Motor.c -o Stepper_Motor.out -lwiringPi
```

Step 4: Run the executable file .

```
sudo ./Stepper_Motor.out
```

As the code runs, the stepper motor first turns clockwise for a round then stop for a second, after that, it rotates anticlockwise for a round; subsequently, the motor stops for a second. This series of actions will be executed repeatedly..

Code

```
#include <stdio.h>

#include <wiringPi.h>
```

23.Stepper_Motor

```
const int motorPins[]={1,4,5,6};       //pins connected to four phase
//1---IN1    4----IN2      5---IN3    6---IN4
const int antiClk[]={0x01,0x02,0x04,0x08};    //right
// 0000 0001      0000    0010      0000 0100    0000 1000
const int clk[]={0x08,0x04,0x02,0x01};      //left
// 0000 1000    0000 0100    0000 0010      0000 0001

void moveSteps(int direction, int steps){
        int step;
        int i=0,j=0;
        for(step=0;step<steps;step++){
                for (j=0;j<4;j++){    //cycle according to power supply order
                for (i=0;i<4;i++){    //assign to each pin
                        if(direction == 1)        //clockwise
                                digitalWrite(motorPins[i],antiClk[j] == (1<<i));
                        else                //anticlockwise
                                digitalWrite(motorPins[i],clk[j] == (1<<i));
                }
                delay(3); //the delay can not be less than 3ms
        }
        }
}
void motorStop(){      //stop rotating
        int i;
        for(i=0;i<4;i++){
                digitalWrite(motorPins[i],LOW); //0000 0000
        }
}
int main(void){
    int i;

    if(wiringPiSetup() == -1){ //when initialize wiring failed,print message to screen
        printf("setup wiringPi failed !");
        return 1;
    }
    for(i=0;i<4;i++){
        pinMode(motorPins[i],OUTPUT);
    }

    while(1){
        moveSteps(1,512); // clock 512 step
        delay(1000);   //delay 1000
        moveSteps(0,512);   // anticlock 512 step
        delay(1000);   //delay 1000
    }
    return 0;
}
```

Code Explanation

```
const int antiClk[]={0x01,0x02,0x04,0x08};   //right
const int clk[]={0x08,0x04,0x02,0x01};     //left
```

These two values are used to indicate the rotation of the Stepper Motor, and the values of them are 0001,0010, 0100, and 1000 respectively that correspond to the electrification condition of four-phase motor when it rotates.

```
for (j=0;j<4;j++){   //cycle according to power supply order
        for (i=0;i<4;i++){   //assign to each pin
                digitalWrite(motorPins[i],antiClk[j] == (1<<i));
        }
```

```
    }
```

This two-layer cycle means that the stepper motor runs for a complete cycle (four steps of the four-phase stepper motor is one cycle).

```
void moveSteps(int direction, int steps){
    int step;
    int i=0,j=0;
    for(step=0;step<steps;step++){
        for (j=0;j<4;j++){    //cycle according to power supply order
        for (i=0;i<4;i++){    //assign to each pin
            if(direction == 1)        //clockwise
                digitalWrite(motorPins[i],antiClk[j] == (1<<i));
            else              //anticlockwise
                digitalWrite(motorPins[i],clk[j] == (1<<i));
        }
        delay(3); //the delay can not be less than 3ms
    }
    }
}
```

This function is used to control the work of the stepper motor, the variable “direction” is used to determine the rotation direction, and the variable “steps” is used to determine the times of the work of stepper motor.

```
        moveSteps(1,512);
```

It takes 2048 steps for the output shaft to turn a circle, and 2048/4=512 times for the stepper motor to work.

For Python Language Users

Step 2: Get into the folder of the code.

```
cd /home/pi/REXQualis_Raspberry_Pi_Complete_Starter_Kit/Python
```

Step 3: Run the executable file.

```
sudo python3 23.Stepper_Motor.py
```

As the code runs, the stepper motor first turns clockwise for a round then stop for a second, after that, it rotates anticlockwise for a round; subsequently, the motor stops for a second. This series of actions will be executed repeatedly.

Code

The code here is for Python3, if you need for Python2, please open the code with the suffix py2 in the attachment.

```
#!/usr/bin/env python3

import RPi.GPIO as GPIO
import time

motorPins = (12, 16, 18, 22)       #pins connected to four phase
antiClk = (0x01,0x02,0x04,0x08)
clk = (0x08,0x04,0x02,0x01)

def setup():
	GPIO.setmode(GPIO.BOARD)            # Numbers GPIOs by physical location
	for pin in motorPins:
		GPIO.setup(pin,GPIO.OUT)



def moveSteps(direction, steps):
	for step in range(steps):
		for j in range(0,4,1):    #cycle according to power supply order
			for i in range(0,4,1):    #assign to each pin
				if (direction == 1):#clockwise
					GPIO.output(motorPins[i],((antiClk[j] == 1<<i) and 
GPIO.HIGH or GPIO.LOW))
```

```
                    else :                          #anticlockwise
                            GPIO.output(motorPins[i],((clk[j] == 1<<i) and
GPIO.HIGH or GPIO.LOW))
                time.sleep(0.003)        #the delay can not be less than 3ms

#function used to stop rotating
def motorStop():
    for i in range(0,4,1):
        GPIO.output(motorPins[i],GPIO.LOW)

def loop():
    while True:
        moveSteps(1,512)
        time.sleep(1)
        moveSteps(0,512)
        time.sleep(1)

def destroy():
    GPIO.cleanup()                    # Release resource

if __name__ == '__main__':          # Program start from here
    setup()
    try:
        loop()
    except KeyboardInterrupt:    # When 'Ctrl+C' is pressed, the program destroy()
will be    executed.
        destroy()
```

Code Explanation

```
antiClk = (0x01,0x02,0x04,0x08)
```

```
clk = (0x08,0x04,0x02,0x01)
```

These two values are used to indicate the rotation of the Stepper Motor, and the values of them are 0001,0010, 0100, and 1000 respectively that correspond to the electrification condition of four-phase motor when it rotates.

```
 for j in range(0,4,1):    #cycle according to power supply order
                for i in range(0,4,1):    #assign to each pin
                        GPIO.output(motorPins[i],((antiClk[j] == 1<<i) and 
GPIO.HIGH or GPIO.LOW))
```

This two-layer cycle means that the stepper motor runs for a complete cycle (four steps of the four-phase stepper motor is one cycle).

```
def moveSteps(direction, steps):
    for step in range(steps):
        for j in range(0,4,1):    #cycle according to power supply order
            for i in range(0,4,1):    #assign to each pin
                if (direction == 1):#clockwise
                    GPIO.output(motorPins[i],((antiClk[j] == 1<<i) and 
GPIO.HIGH or GPIO.LOW))
                else :                    #anticlockwise
                    GPIO.output(motorPins[i],((clk[j] == 1<<i) and 
GPIO.HIGH or GPIO.LOW))
            time.sleep(0.003)      #the delay can not be less than 3ms
```

This function is used to control the work of the stepper motor, in which direction is used to determine the rotation direction, and steps are used to determine the times of the work of stepper motor.

moveSteps(1,512)

It takes 2048 steps for the output shaft to turn a circle, and 2048/4=512 times for the stepper motor to work.

Phenomenon Picture