Engineering Innovation Week

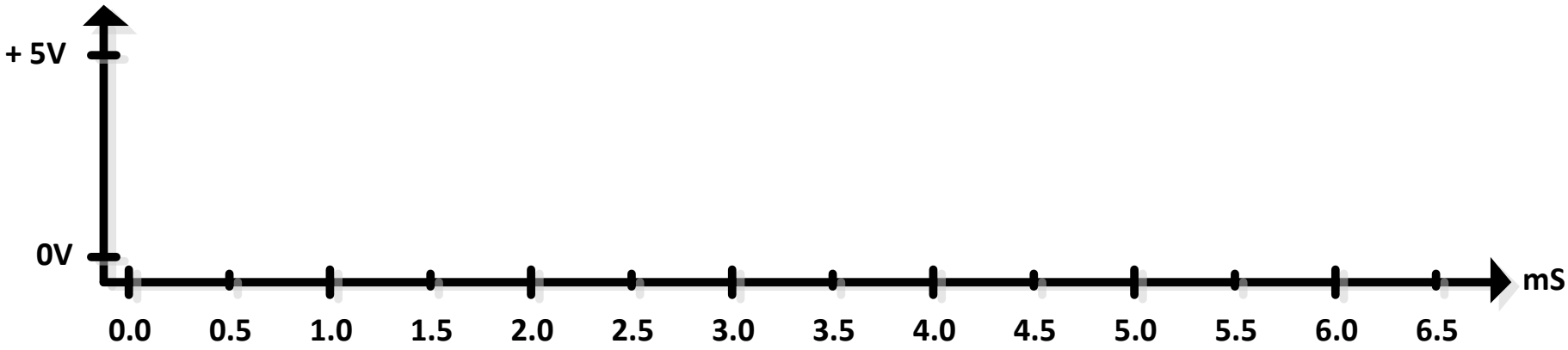
University of Johannesburg (23 June – 27 June 2014)

Controlling Devices

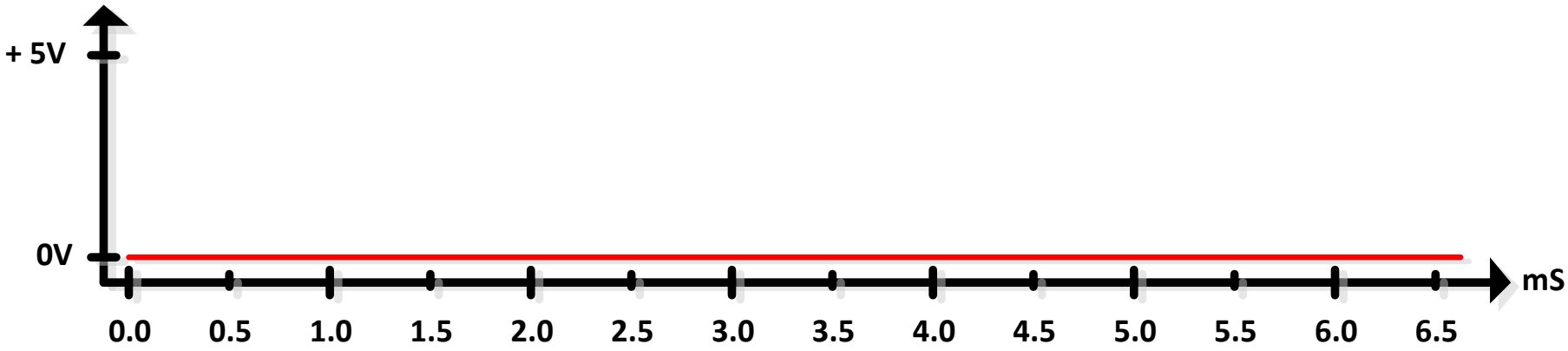
intel Galileo

What will you make?

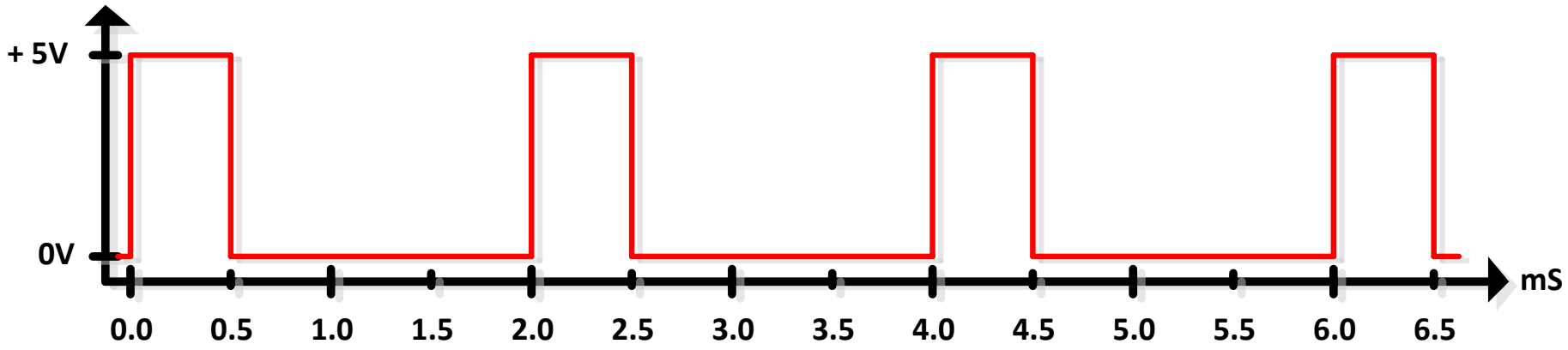
Plotting Along X and Y



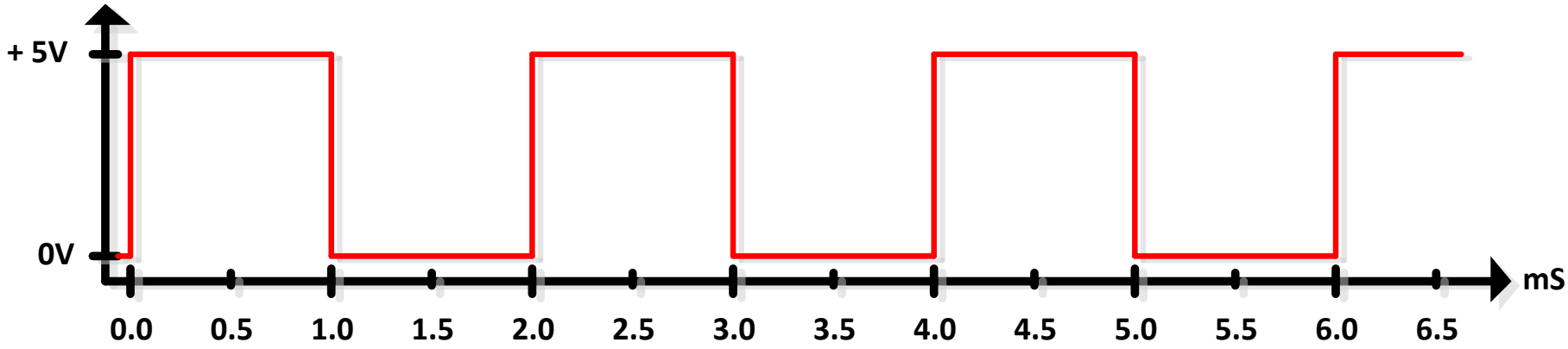
Pulse Width Modulation, 0% Power



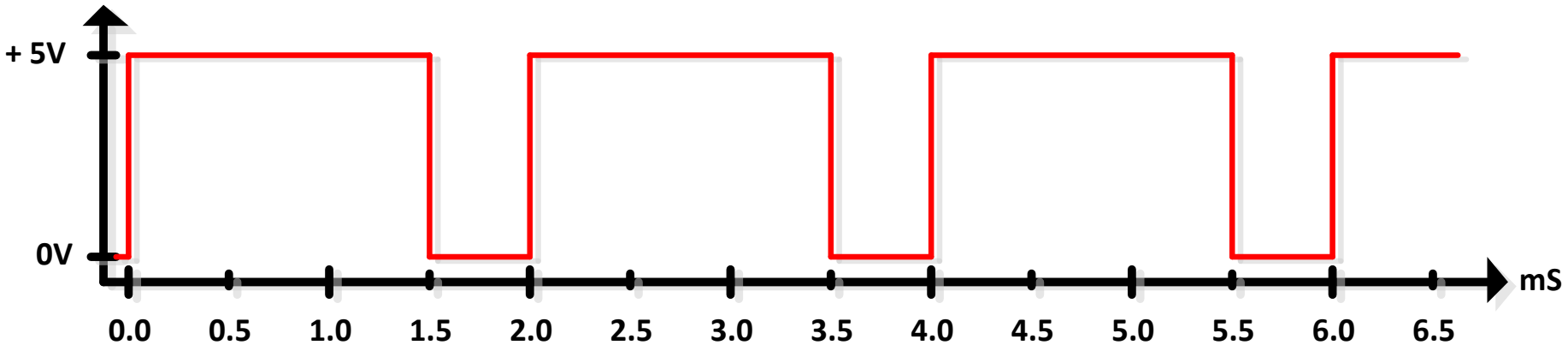
Pulse Width Modulation, 25% Power



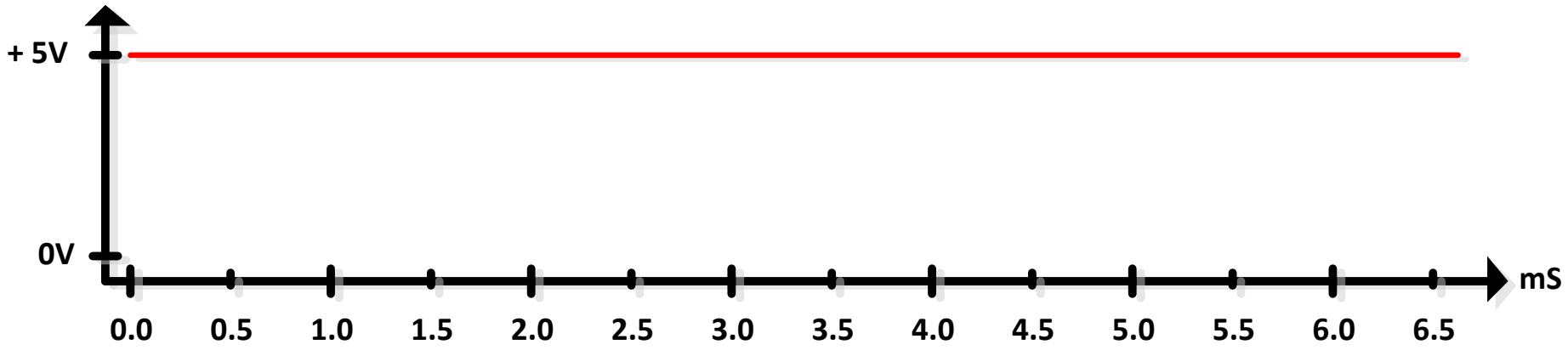
Pulse Width Modulation, 50% Power



Pulse Width Modulation, 75% Power



Pulse Width Modulation, 100% Power



PWM with the Analog Write Library

PWM frequency is about 490 Hz

analogWrite function has nothing to do with the analogRead function

Syntax

analogWrite(pin, value)

Parameters

“pin” is the pin number to write to (only certain pins available; ~)

“value” is the duty-cycle of the PWM waveform

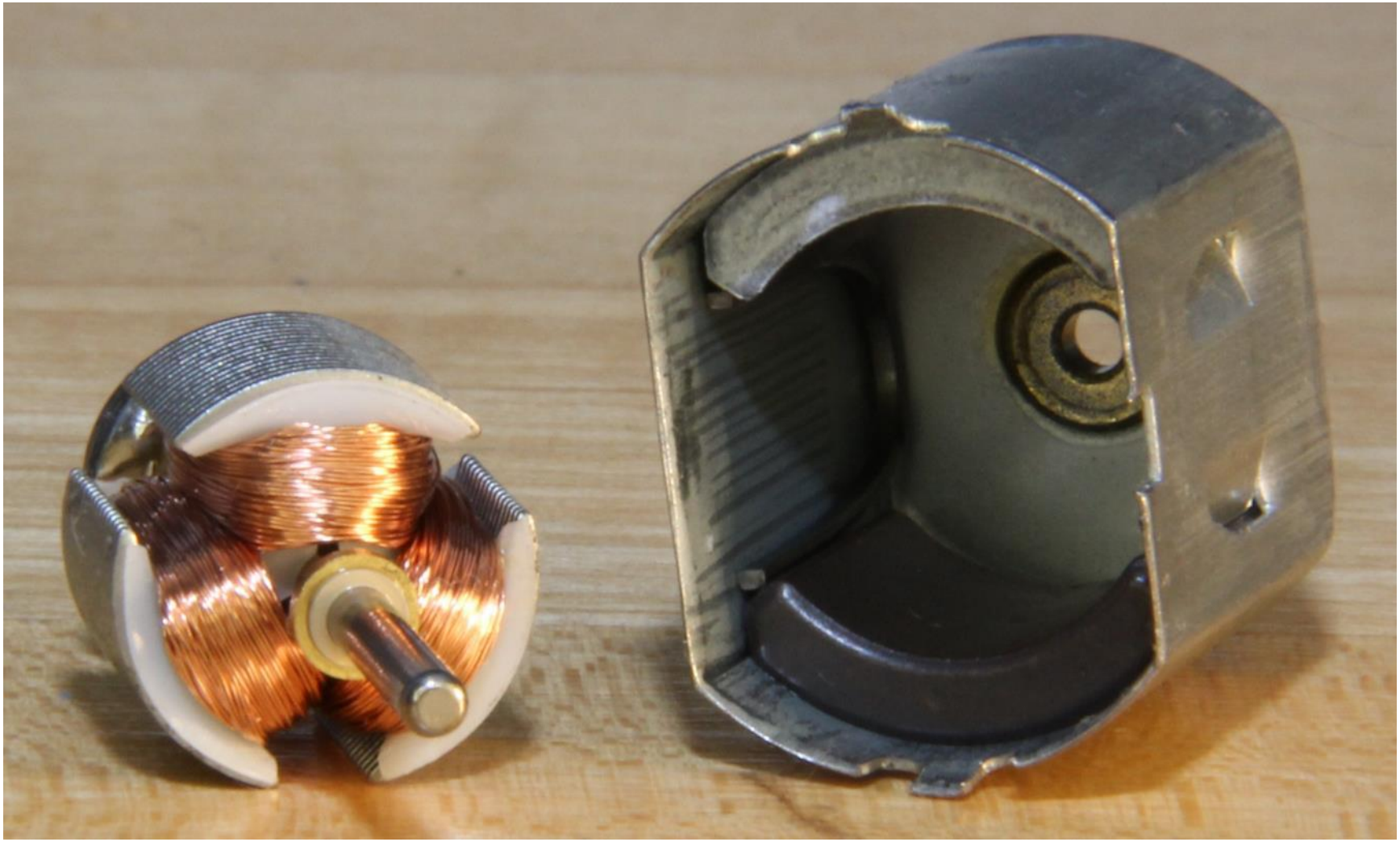
An 8-bit value = 256 distinct settings

“0” = completely off

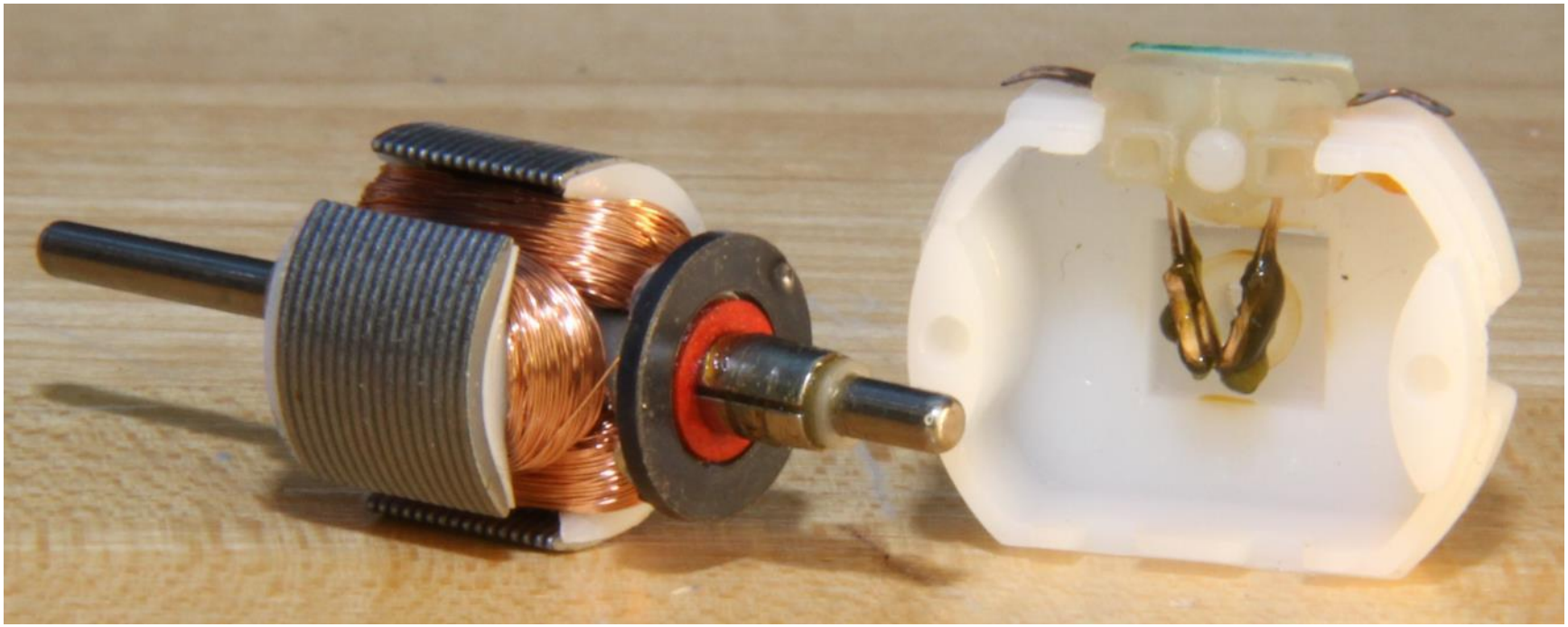
“255” = completely on

Brushed DC Motor Rotor

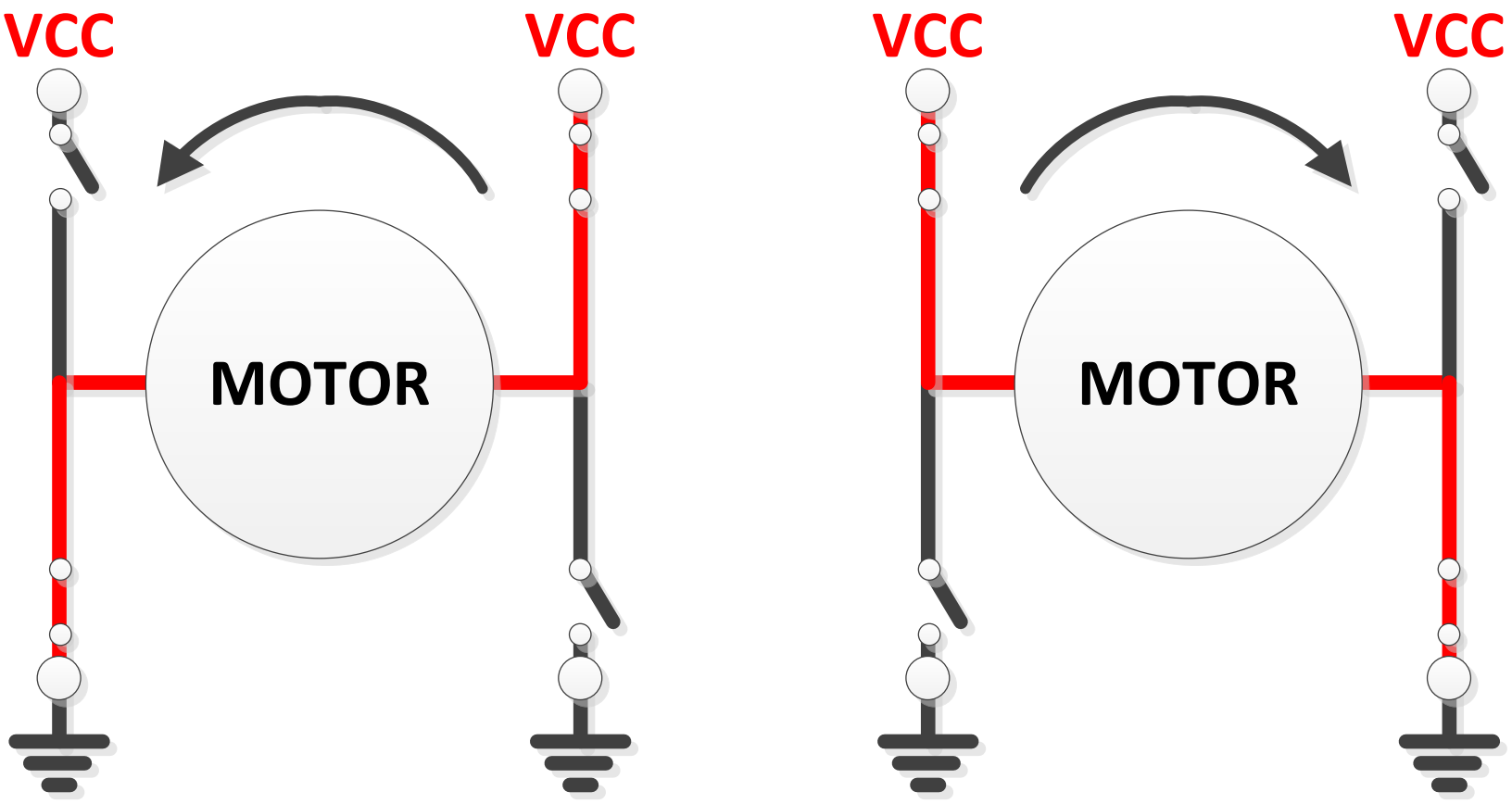
Brushed DC Motor Stator and Rotor



Brushed DC Motor Rotor with Commutator and Brushes



H-Bridge



L293DNE Motor Driver Overview

Package Geometry

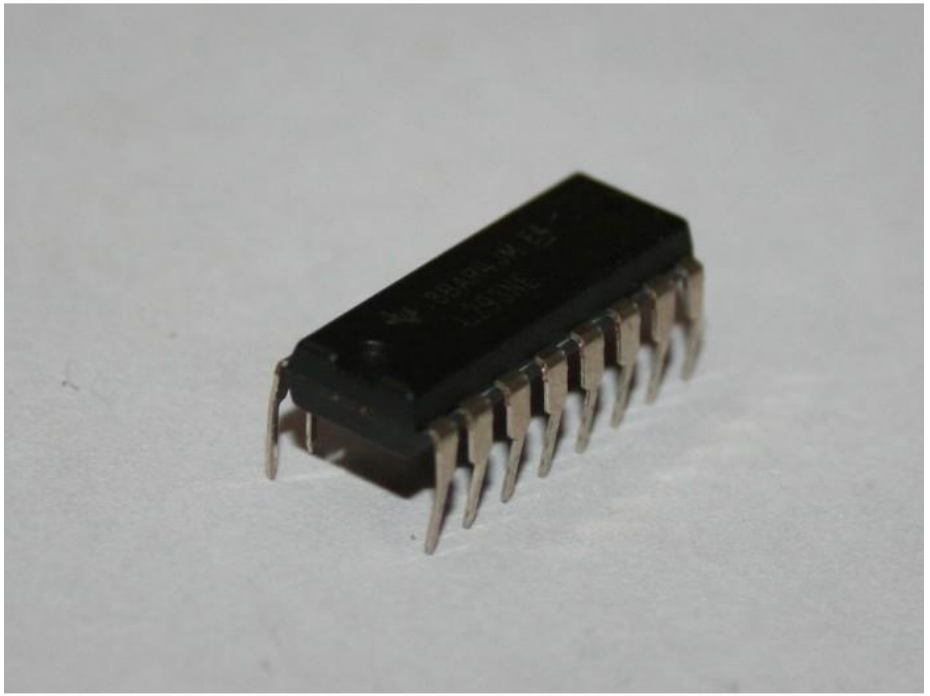
Driver

Pin out

Operation

Software Interface and Control

Break



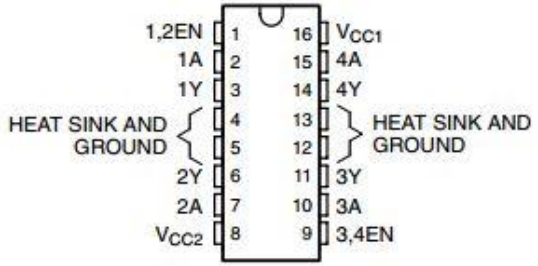
L293DNE Data Sheet

L293, L293D
QUADRUPLE HALF-H DRIVERS

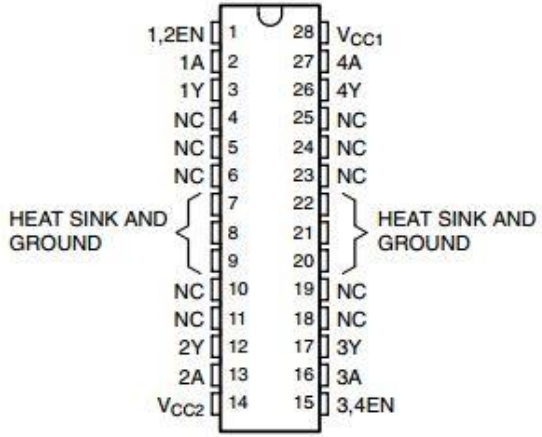
SLRS008C – SEPTEMBER 1986 – REVISED NOVEMBER 2004

- **Featuring Unitrode L293 and L293D Products Now From Texas Instruments**
- **Wide Supply-Voltage Range: 4.5 V to 36 V**
- **Separate Input-Logic Supply**
- **Internal ESD Protection**
- **Thermal Shutdown**
- **High-Noise-Immunity Inputs**
- **Functionally Similar to SGS L293 and SGS L293D**
- **Output Current 1 A Per Channel (600 mA for L293D)**
- **Peak Output Current 2 A Per Channel (1.2 A for L293D)**
- **Output Clamp Diodes for Inductive Transient Suppression (L293D)**

L293 . . . N OR NE PACKAGE
L293D . . . NE PACKAGE
(TOP VIEW)



L293 . . . DWP PACKAGE
(TOP VIEW)

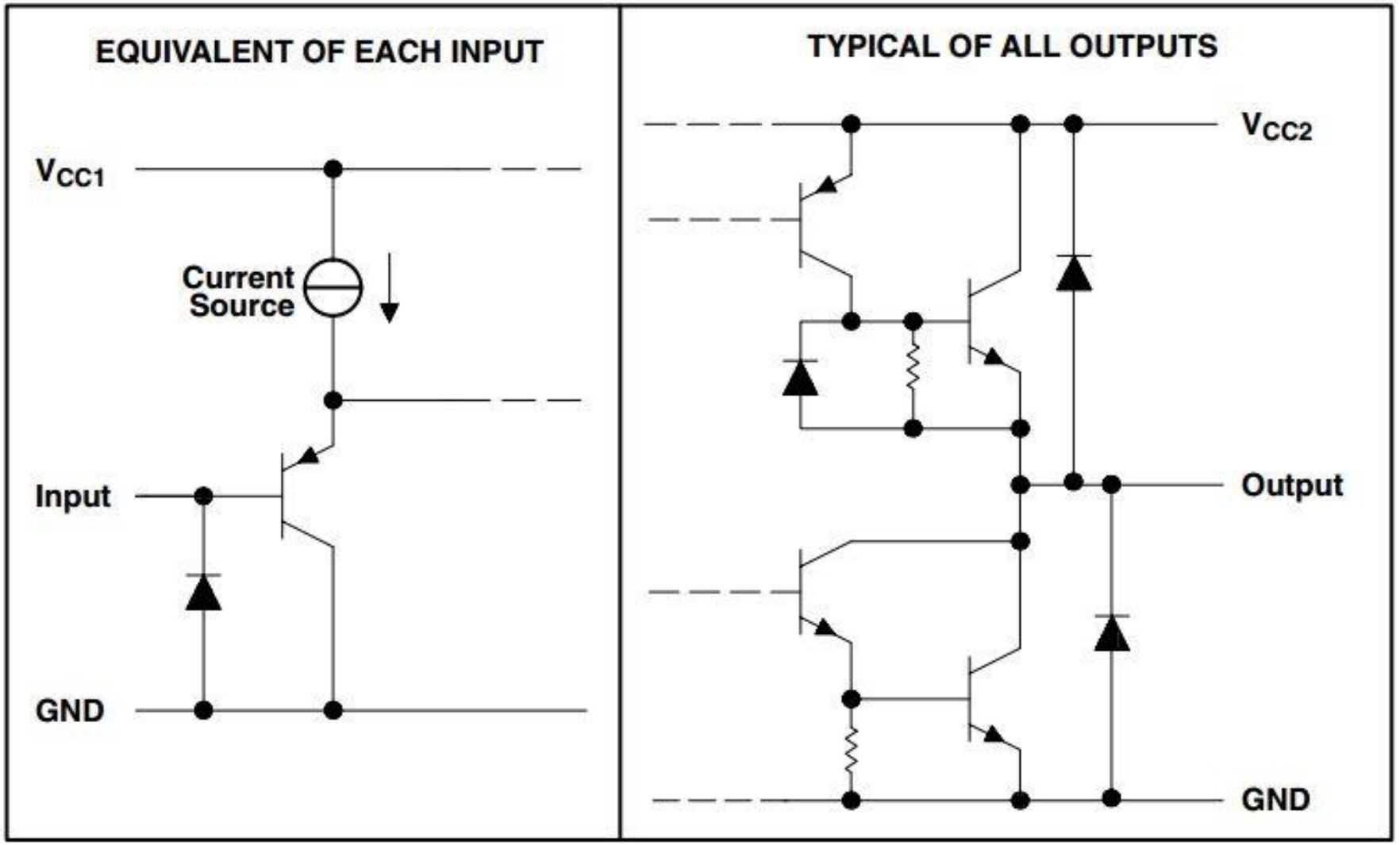


description/ordering information

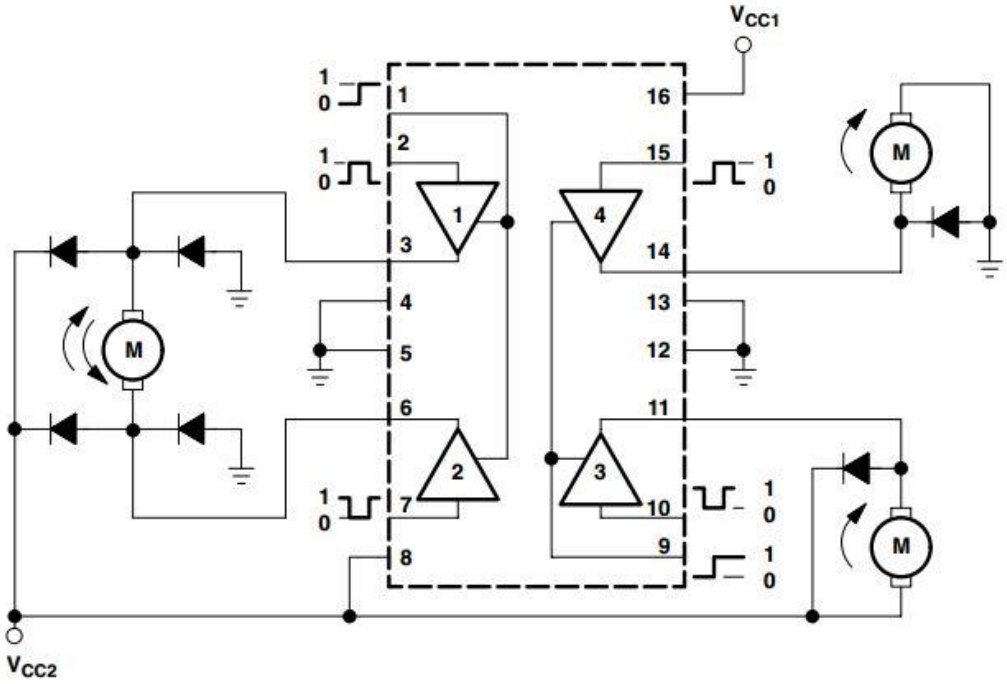
The L293 and L293D are quadruple high-current half-H drivers. The L293 is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays, solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

All inputs are TTL compatible. Each output is a complete totem-pole drive circuit, with a Darlington transistor sink and a pseudo-Darlington source. Drivers are enabled in pairs, with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN. When an enable input is high, the associated drivers are enabled, and their outputs are active and in phase with their inputs. When the enable input is low, those drivers are disabled, and their outputs are off and in the high-impedance state. With the proper data inputs, each pair of drivers forms a full-H (or bridge) reversible drive suitable for solenoid or motor applications.

L293DNE I/O Schematic Representation



L293DNE Block Diagram



NOTE: Output diodes are internal in L293D.

FUNCTION TABLE
(each driver)

INPUTS†		OUTPUT
A	EN	Y
H	H	H
L	H	L
X	L	Z

H = high level, L = low level, X = irrelevant,
Z = high impedance (off)
† In the thermal shutdown mode, the output is in the high-impedance state, regardless of the input levels.

Brushed DC Motor

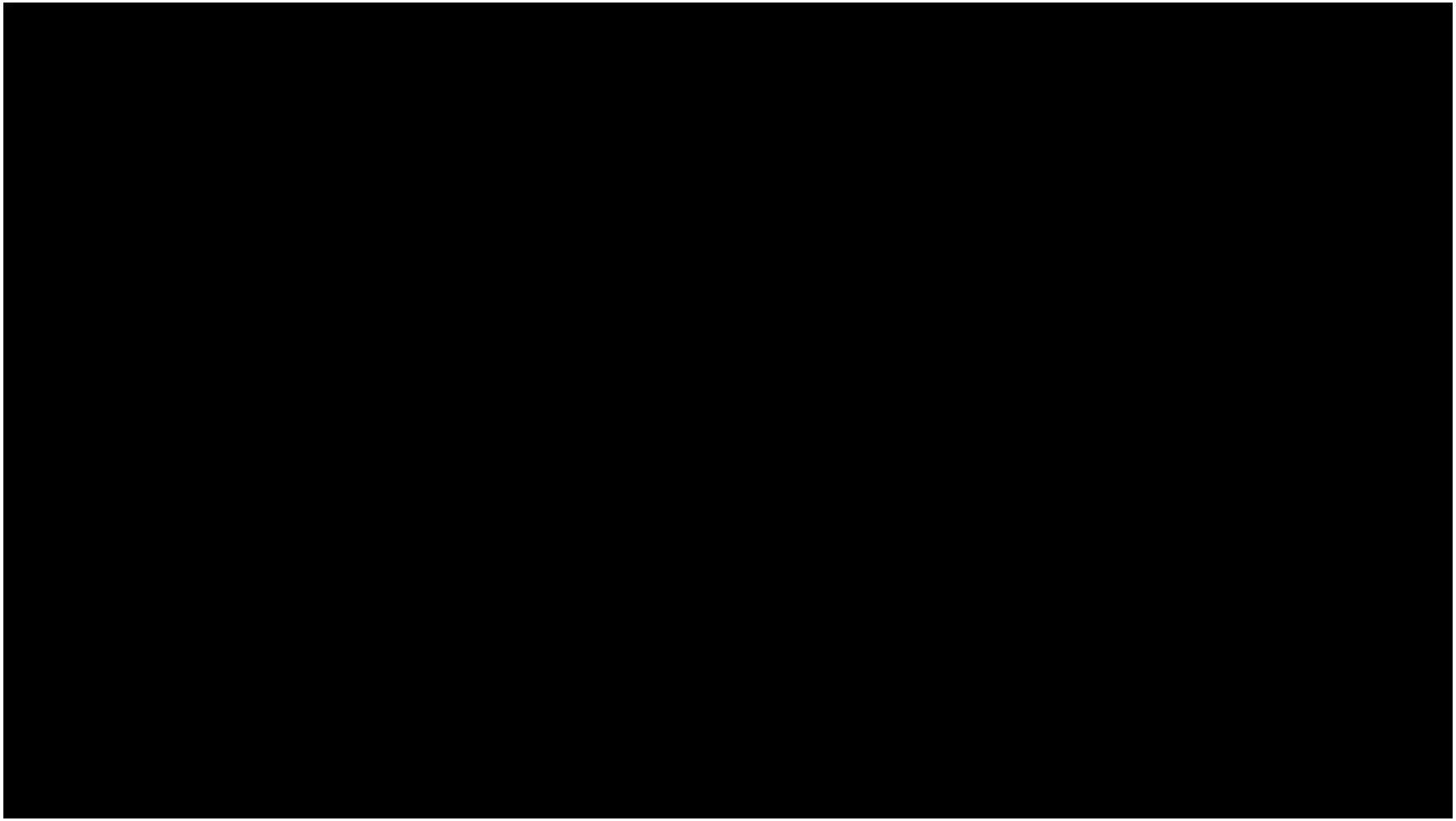
Construction

Driver

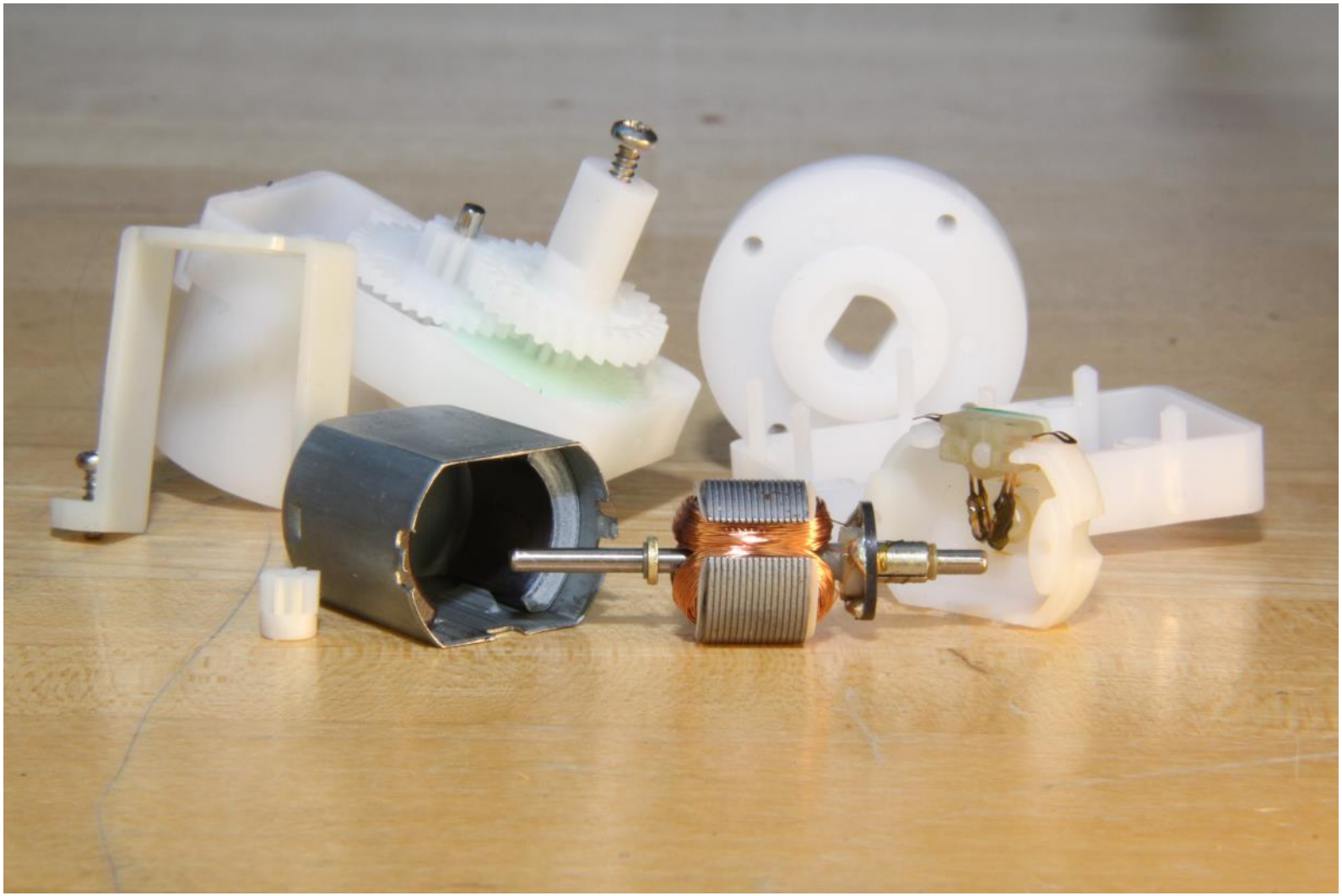
Software Control

LAB

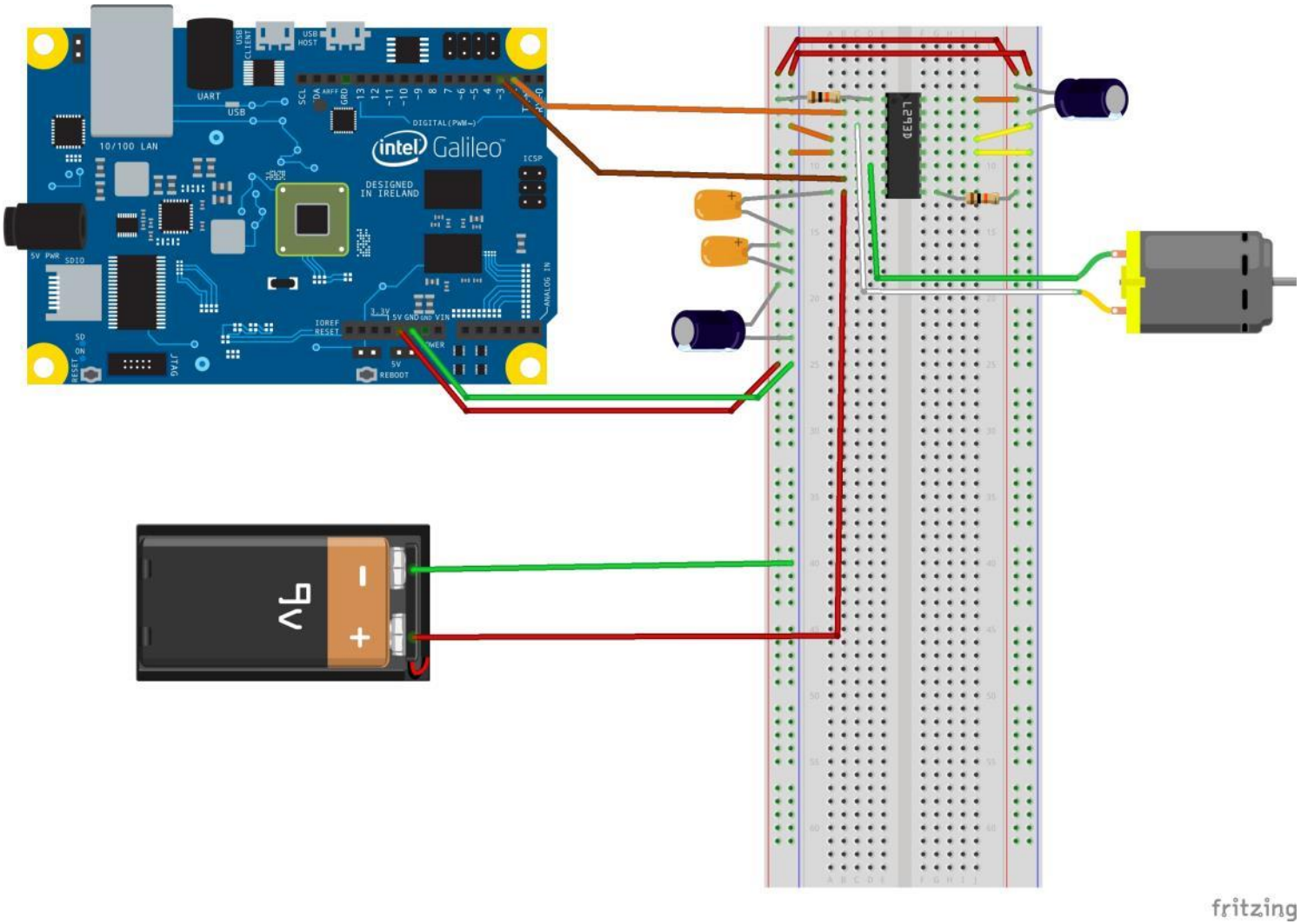
DC Motors, BallBot



Internal Workings of Brushed DC Motor

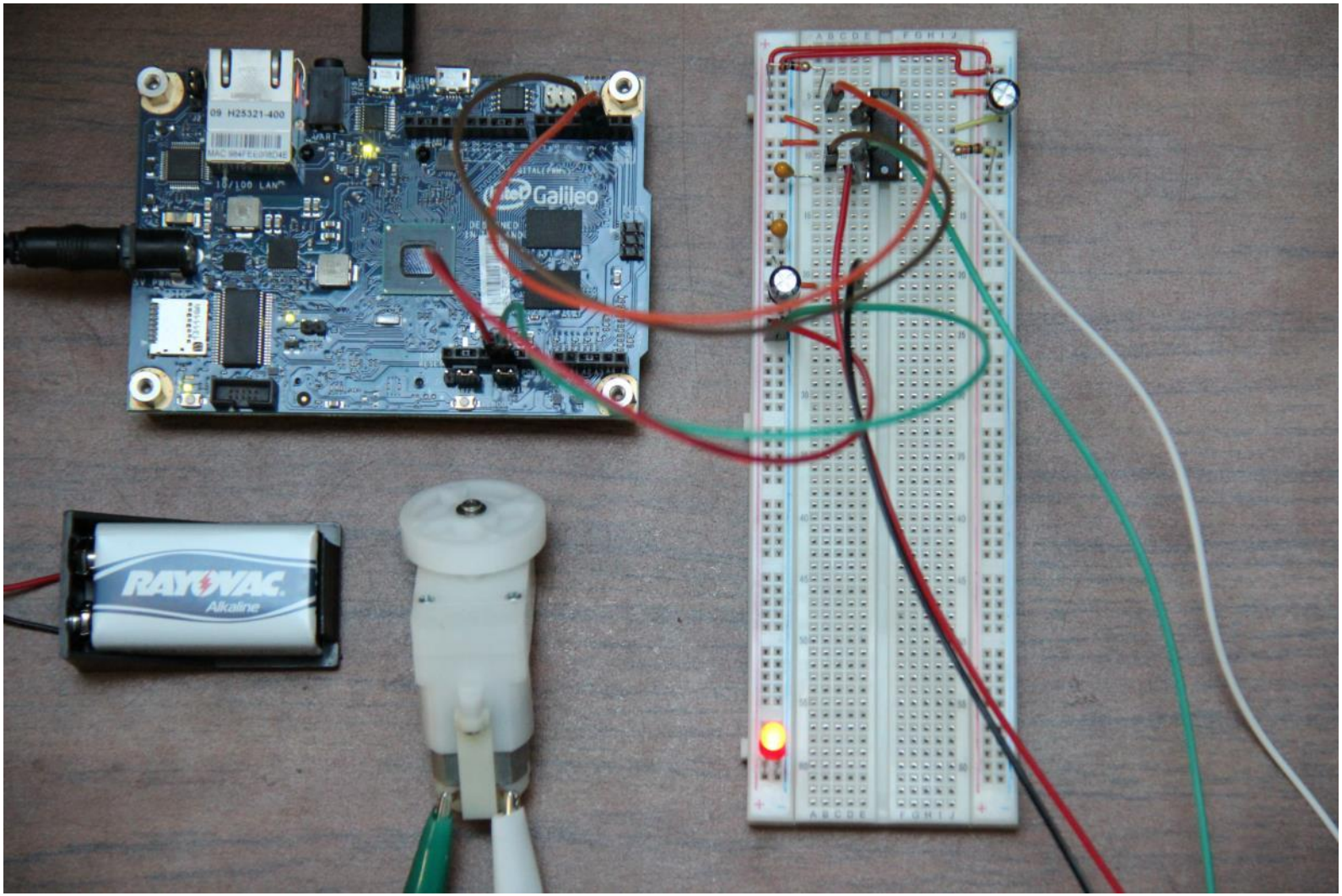


Brushed DC Motor Wire Diagram



WHAT WILL YOU MAKE?

Brushed DC Motor Wiring



Script; Brushed DC Motor

/*

This is basic code to turn on a DC motor through a L293DN motor driver chip. The motor is being driven by pins 2 and 3 on the Galileo board.

Making Pin 2 high and Pin 3 low will rotate the DC motor in one direction. Making Pin 2 low and Pin 2 high will rotate the DC motor in the opposite direction.

*/

Script; Brushed DC Motor –continue-

```
int DC_plus = 3;                    // one leg of the DC motor
int DC_minus = 2;                   // other leg of the DC motor

void setup() {
  pinMode(DC_plus, OUTPUT);       // make DC motor pins outputs
  pinMode(DC_minus, OUTPUT);    // make DC motor pins outputs
}
```

Script; Brushed DC Motor –continue-

```
void loop() {
  digitalWrite(DC_plus, HIGH);                    // make pin 3 high
  digitalWrite(DC_minus, LOW);                    // make pin 2 low

  // this makes the DC motor run in a direction
}
```

DC Motor Speed Control

Can you control the speed of the motor by including Pulse Width Modulation?

LAB

Stepper Motor

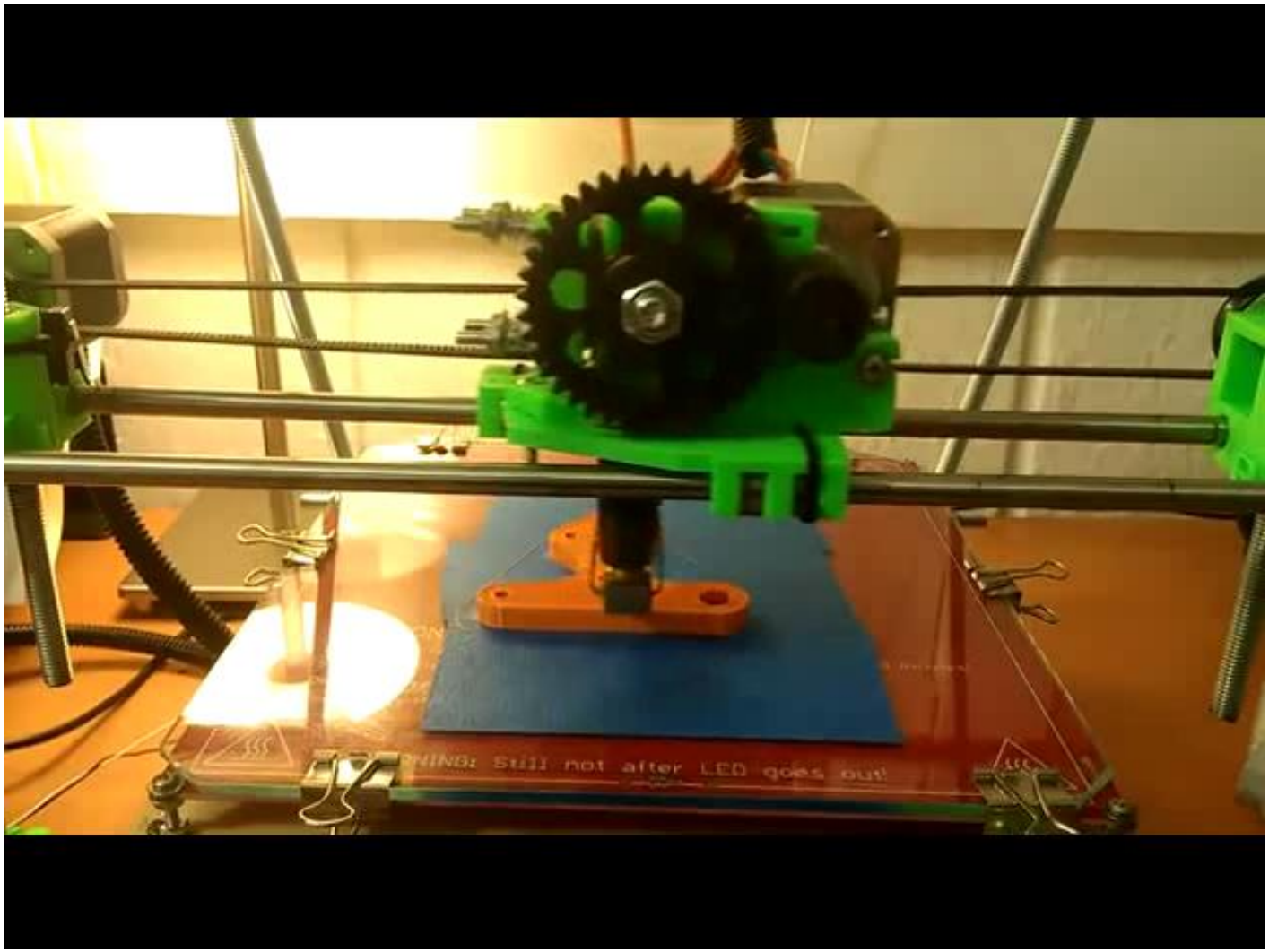
Construction

Driver

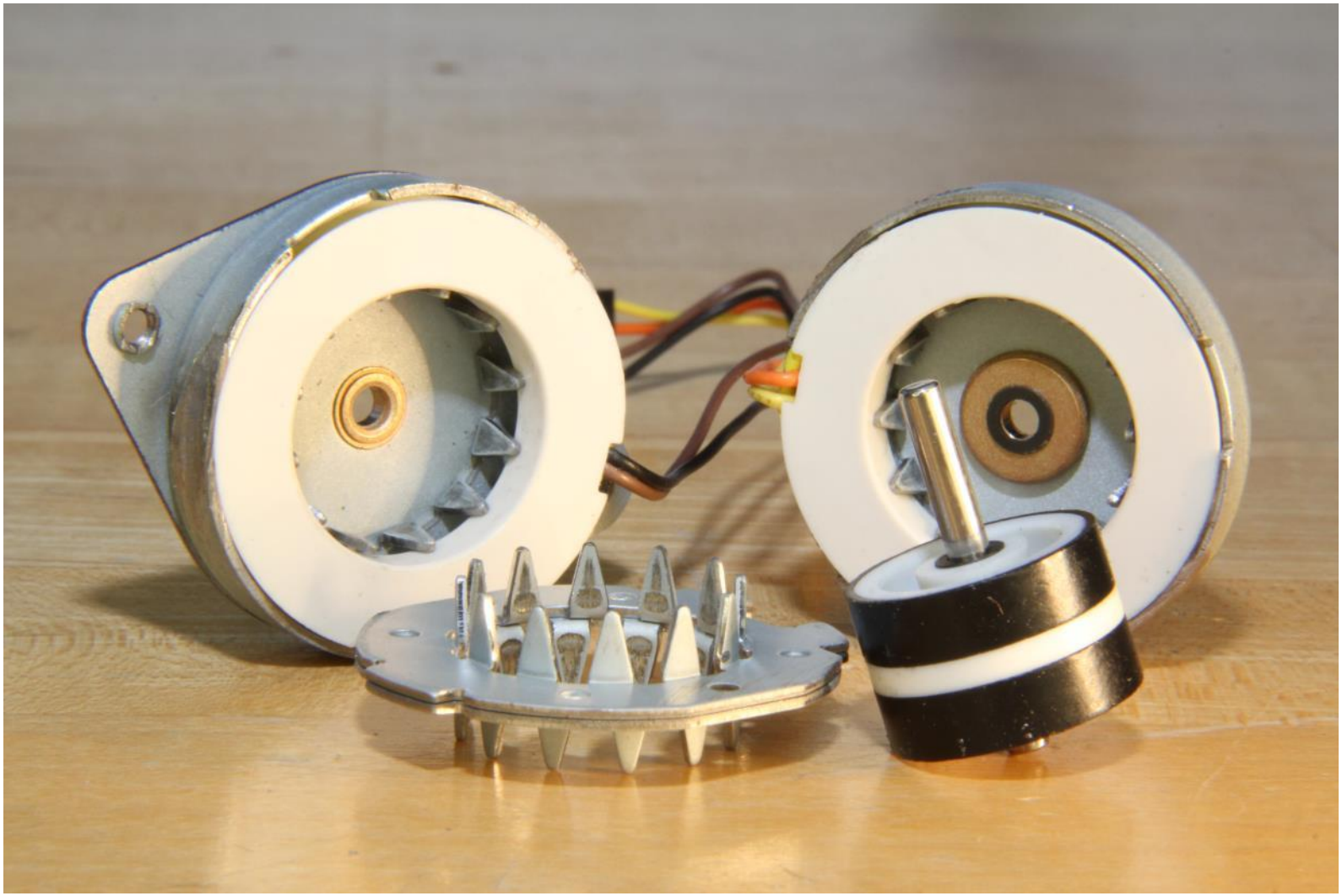
Software Control

LAB

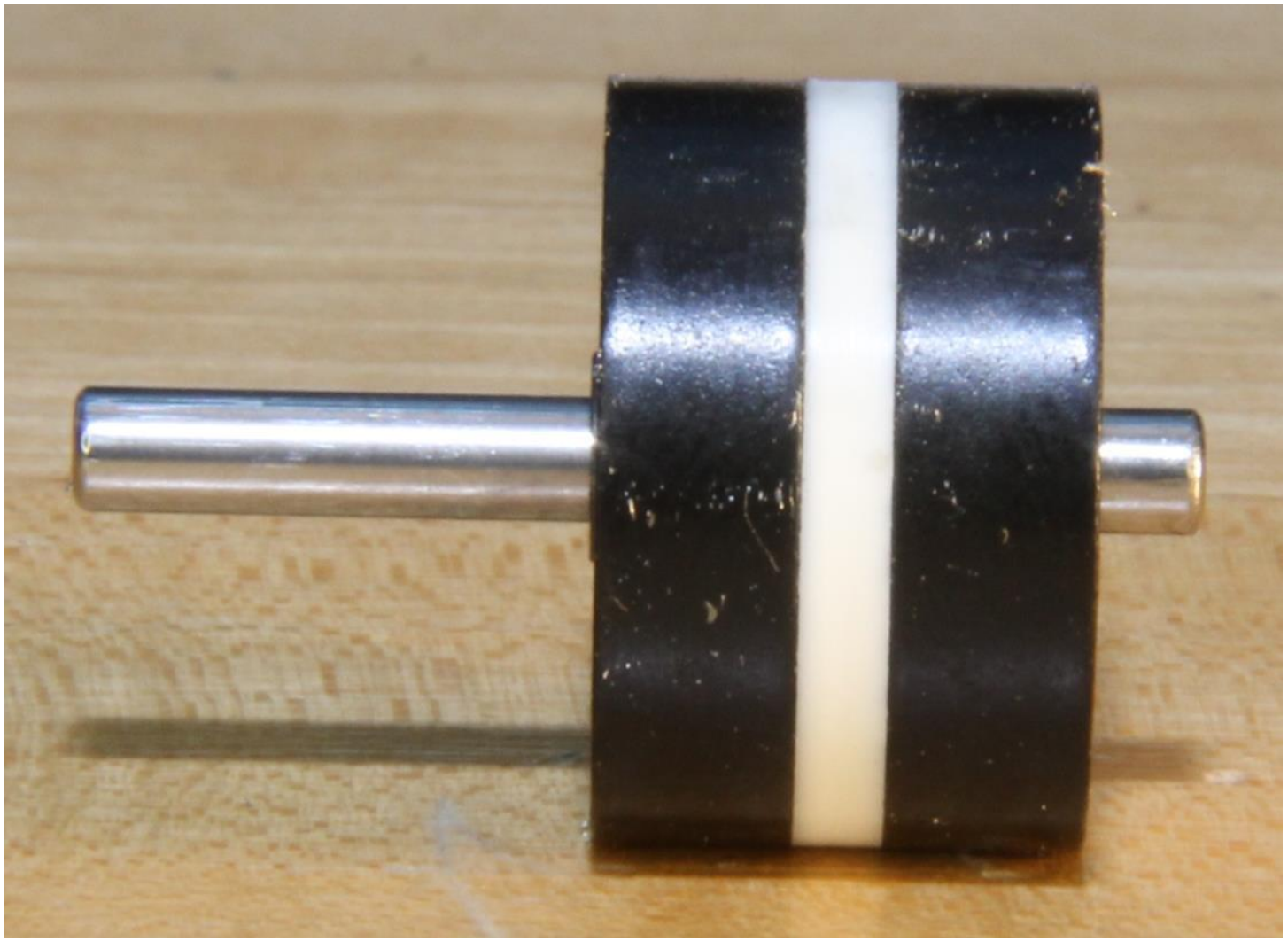
Steppers, 3-D Printing



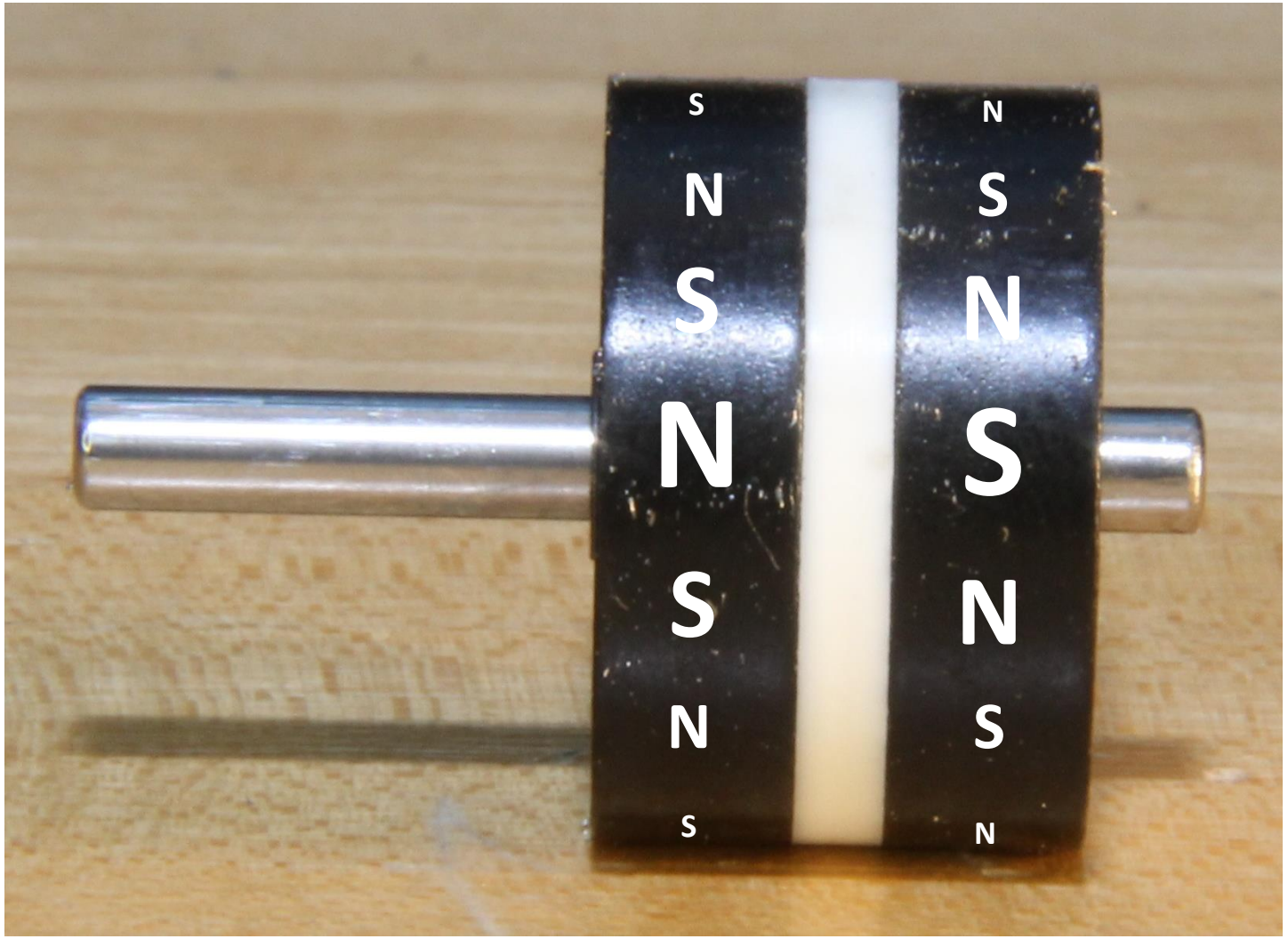
Internal Workings of Stepper Motor



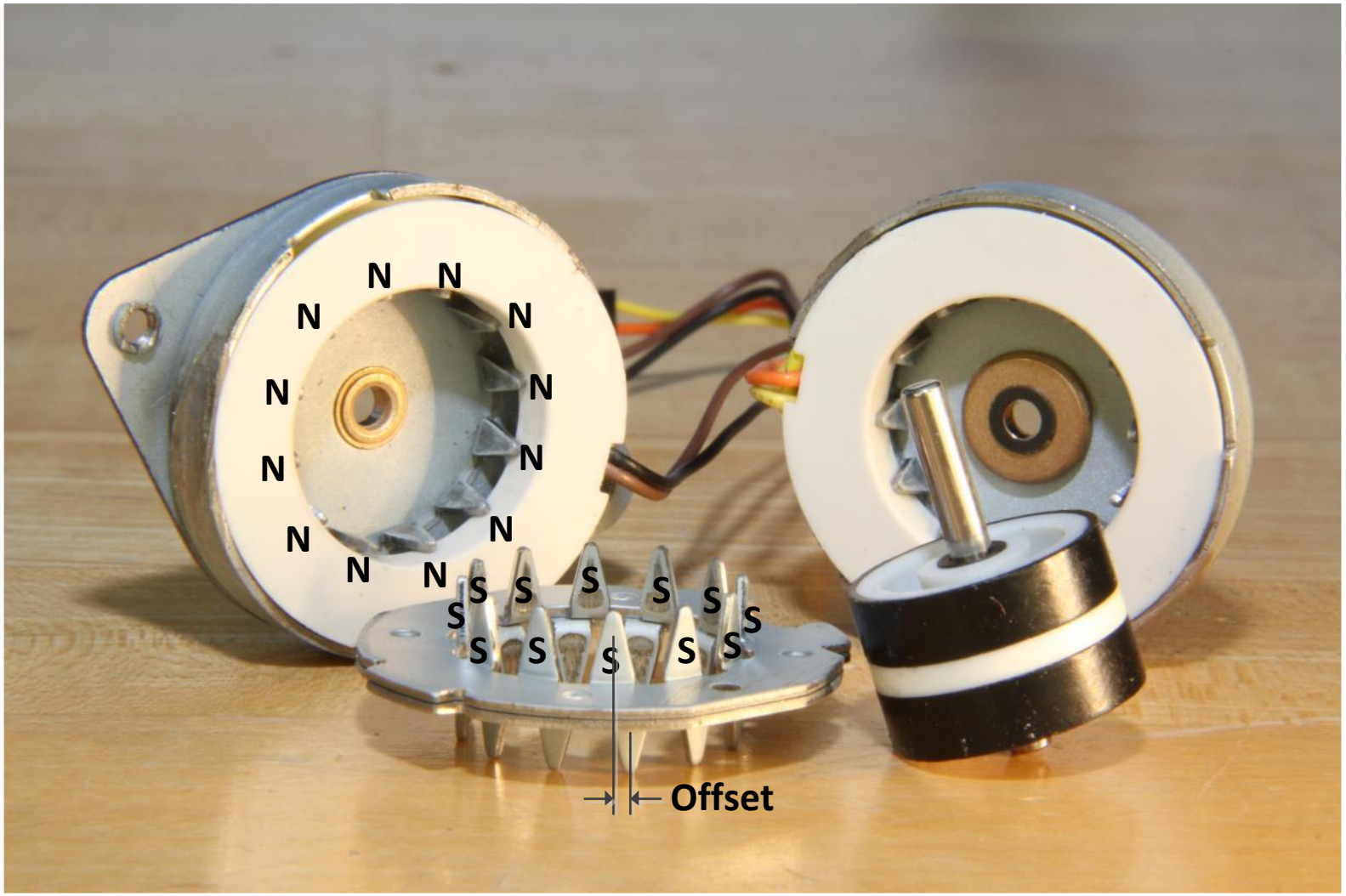
Stepper Motor Rotor



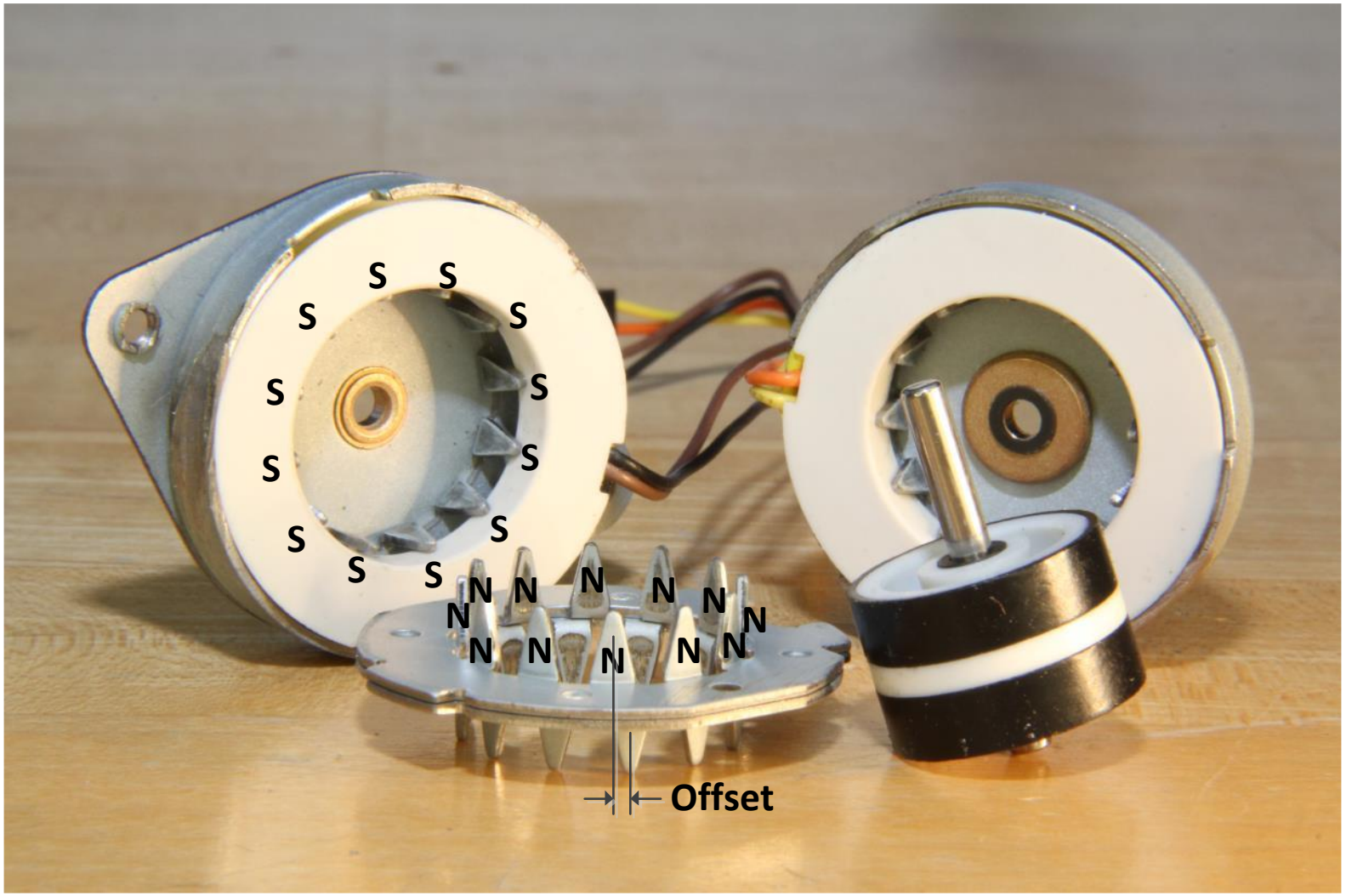
Stepper Motor Rotor



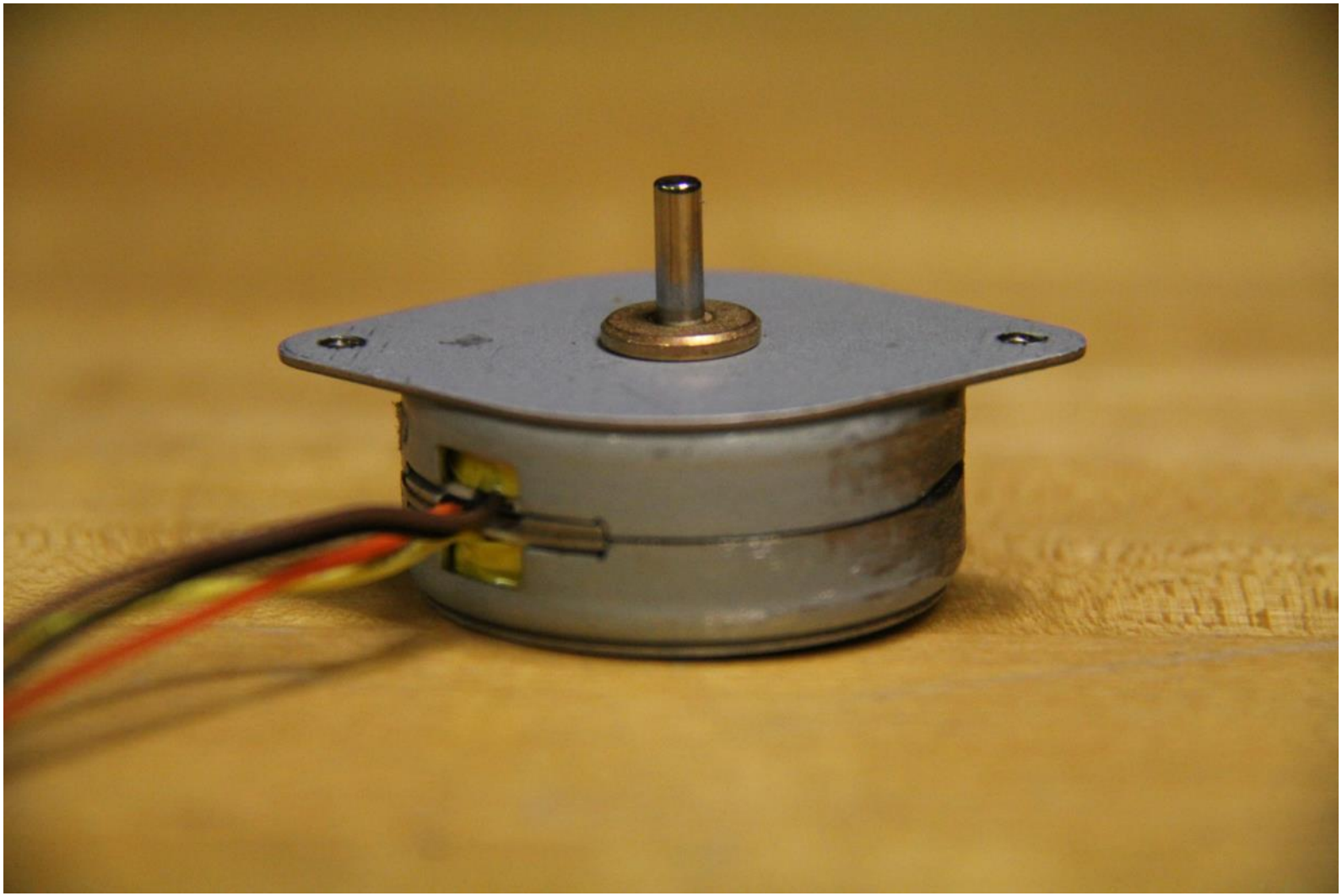
Stepper Motor Stator



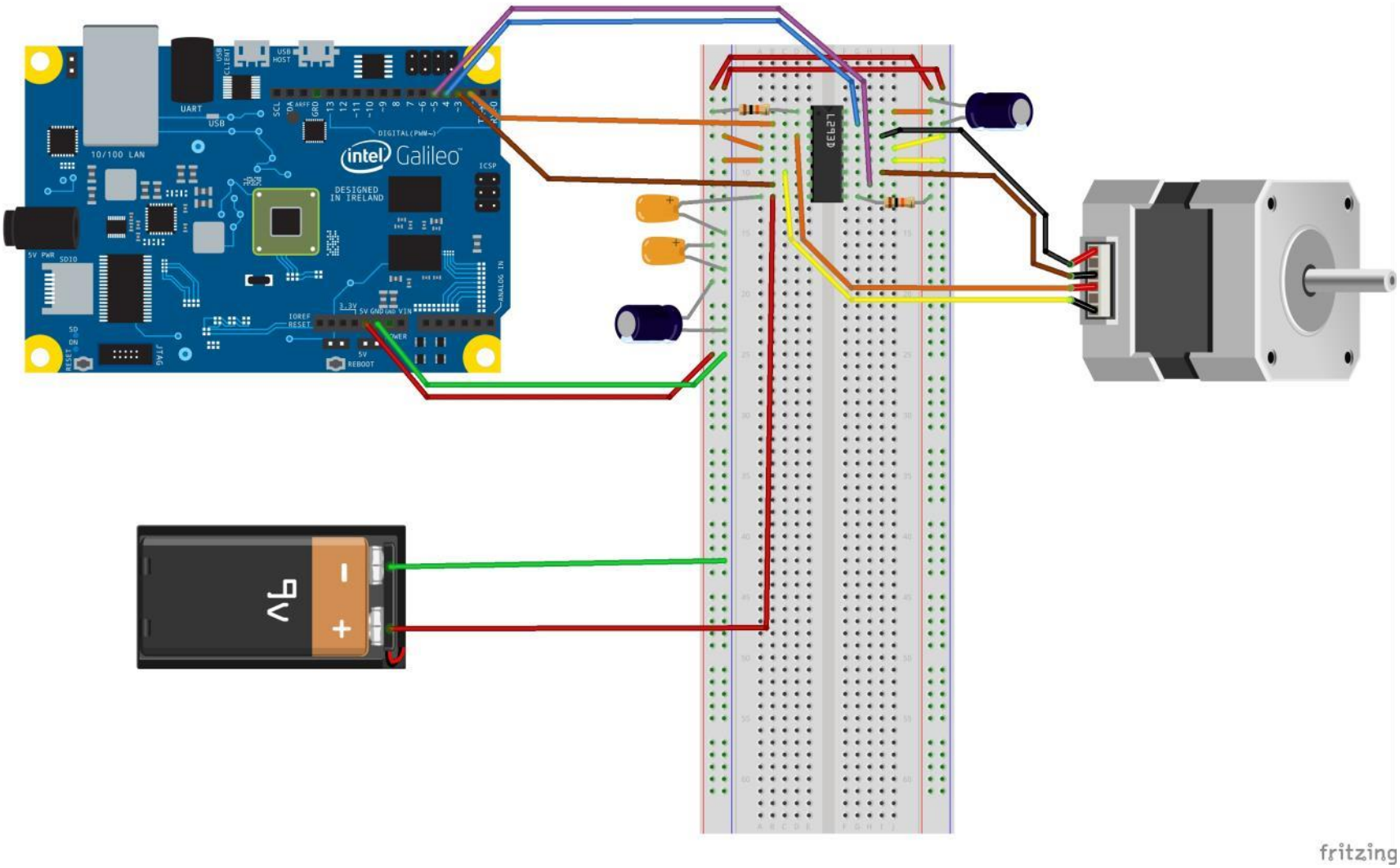
Stepper Motor Stator



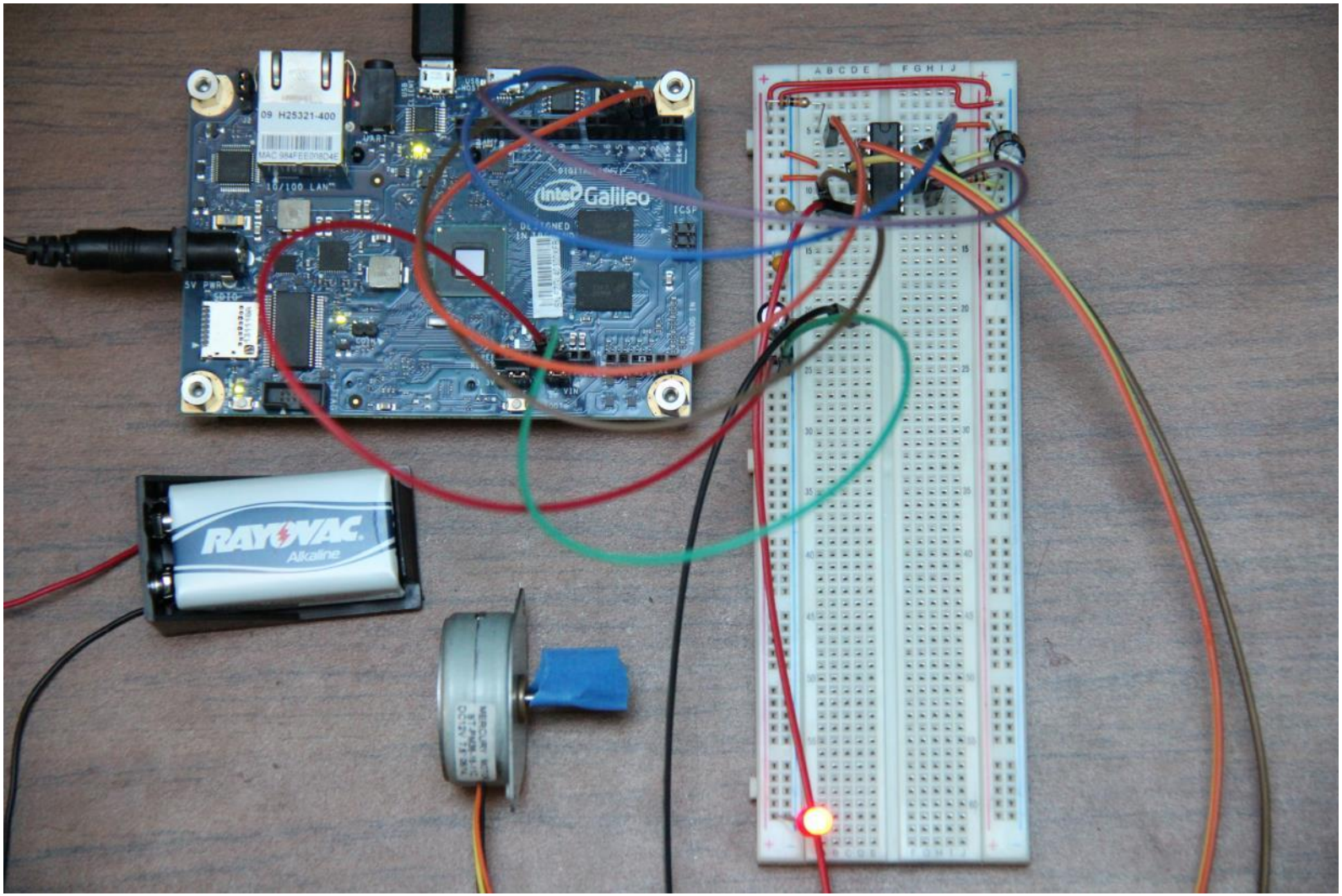
Stepper Motor



Stepper Motor Wire Diagram



Stepper Motor Wiring



Stepper Library

We are utilizing a 4-wire, bi-polar stepper motor configuration

First we configure the motor

Syntax

Stepper(steps, pin1, pin2, pin3, pin4)

Parameters

“steps” is the discrete number of steps needed for one revolution

7.5° per step; (360° / 7.5°) = 48 steps per revolution

“pin1” the first of four pins attached to the motor

“pin2” the second of four pins attached to the motor

“pin3” the third of four pins attached to the motor

“pin4” the fourth of four pins attached to the motor

Stepper Library (continue)

Now to control the motor

Syntax

setSpeed(rpm)

Parameters

“rpm” is the speed in revolutions per minute

Note: this doesn’t make the motor turn, just sets the speed

Stepper Library (continue)

Syntax

Step(steps)

Parameters

“steps” is the number of discrete steps you want the motor to make

a positive number rotates in one direction,

a negative number rotates in the opposite direction.

Note: executes these steps at a speed determined by the most recent call to setSpeed().

Attention: this function is ***blocking*** (nothing else happens in your code until this action is complete)

Script; Basic Stepper Motor

/*

This is basic code to turn on a Bipolar Stepper motor using a L293DN motor driver chip and the "Stepper Library". One of the bipolar stepper motor coils is driven from pins 2 and 3 on the Galileo board. The second bipolar stepper motor coil is driven from pins 4 and 5 on the Galileo board.

We are using the "Stepper Library", the L293D driver, and a Small Stepper Motor from Sparkfun; Part-Number ROB-10551.

Stepper Motor Wires:

Yellow & Orange Coil 1

Black & Brown Coil 2

*/

Script; Basic Stepper Motor –continue-

```
#include <Stepper.h>

int Pin_01 = 2;
int Pin_02 = 3;
int Pin_03 = 4;
int Pin_04 = 5;

int Number_Of_Steps = 1;
int Step_Speed = 100;

Stepper Motor_01(48, Pin_01, Pin_02, Pin_03, Pin_04);
```

Script; Basic Stepper Motor –continue-

```
void setup() {
  pinMode(Pin_01, OUTPUT);
  pinMode(Pin_02, OUTPUT);
  pinMode(Pin_03, OUTPUT);
  pinMode(Pin_04, OUTPUT);

  Motor_01.setSpeed(Step_Speed);
}
```

Script; Basic Stepper Motor –continue-

```
void loop() {
    Motor_01.step(Number_Of_Steps);
}
```

LAB

Servo Drive

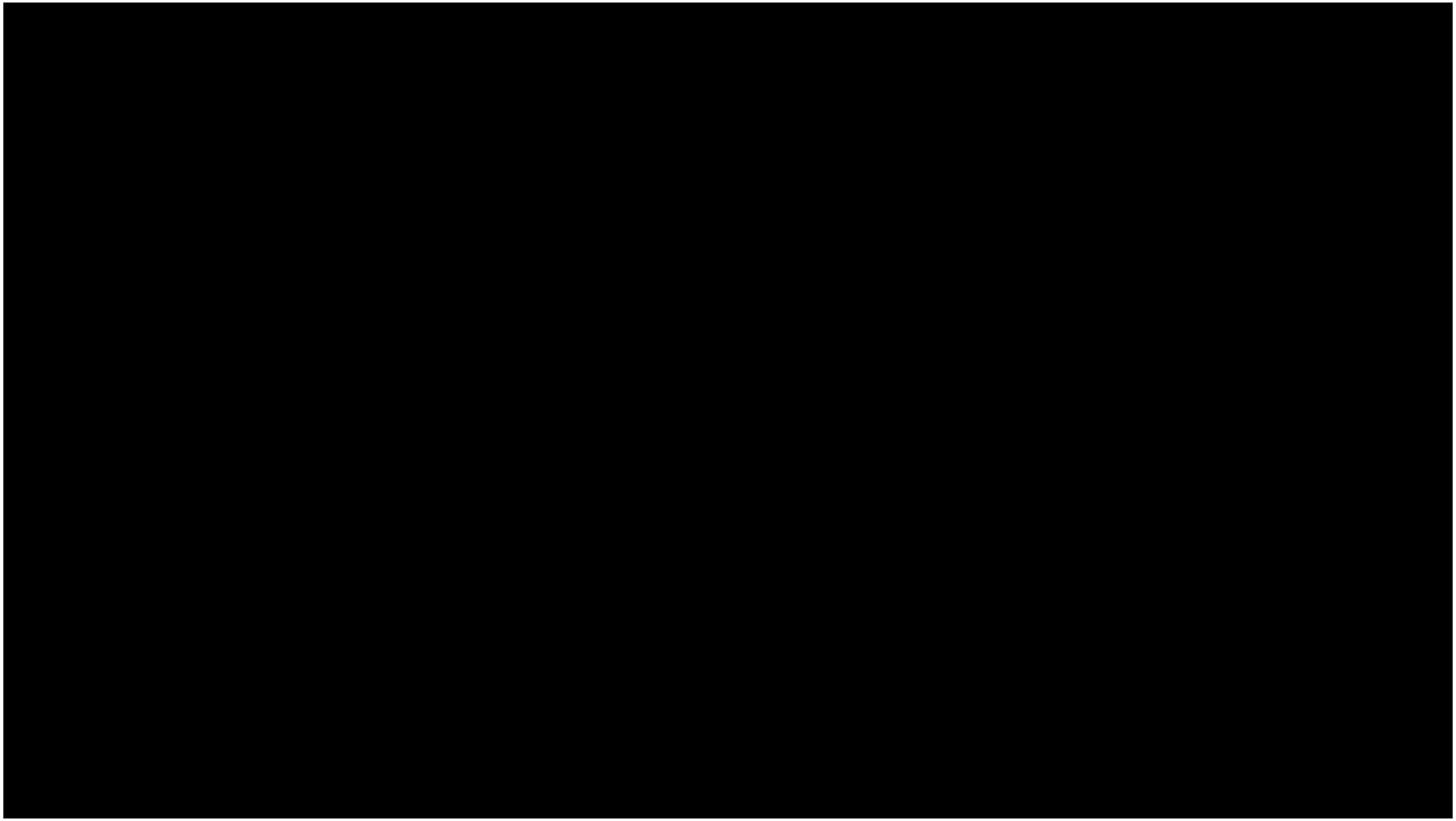
Construction

Driver

Software Control

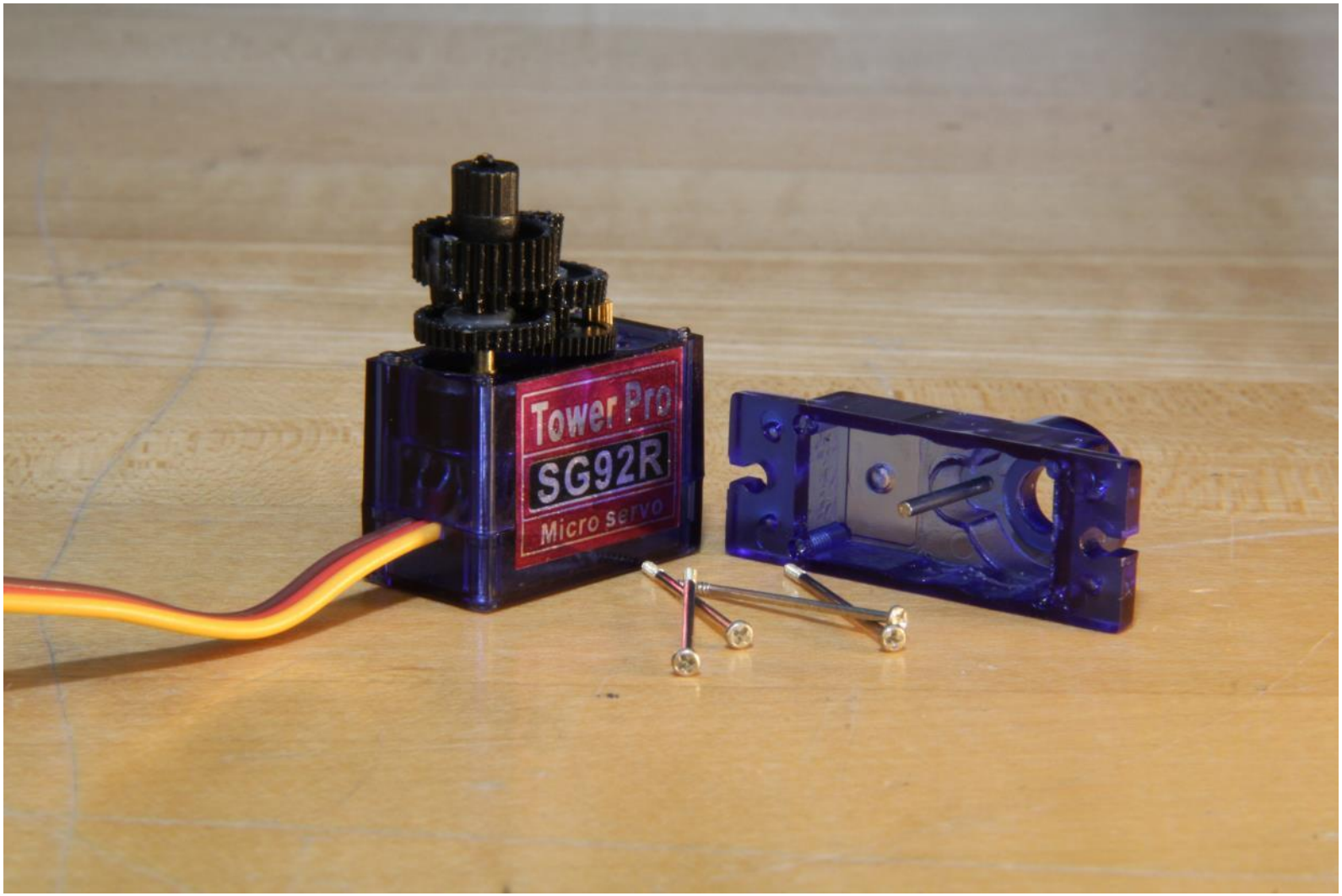
LAB

Servos, Lego Bot

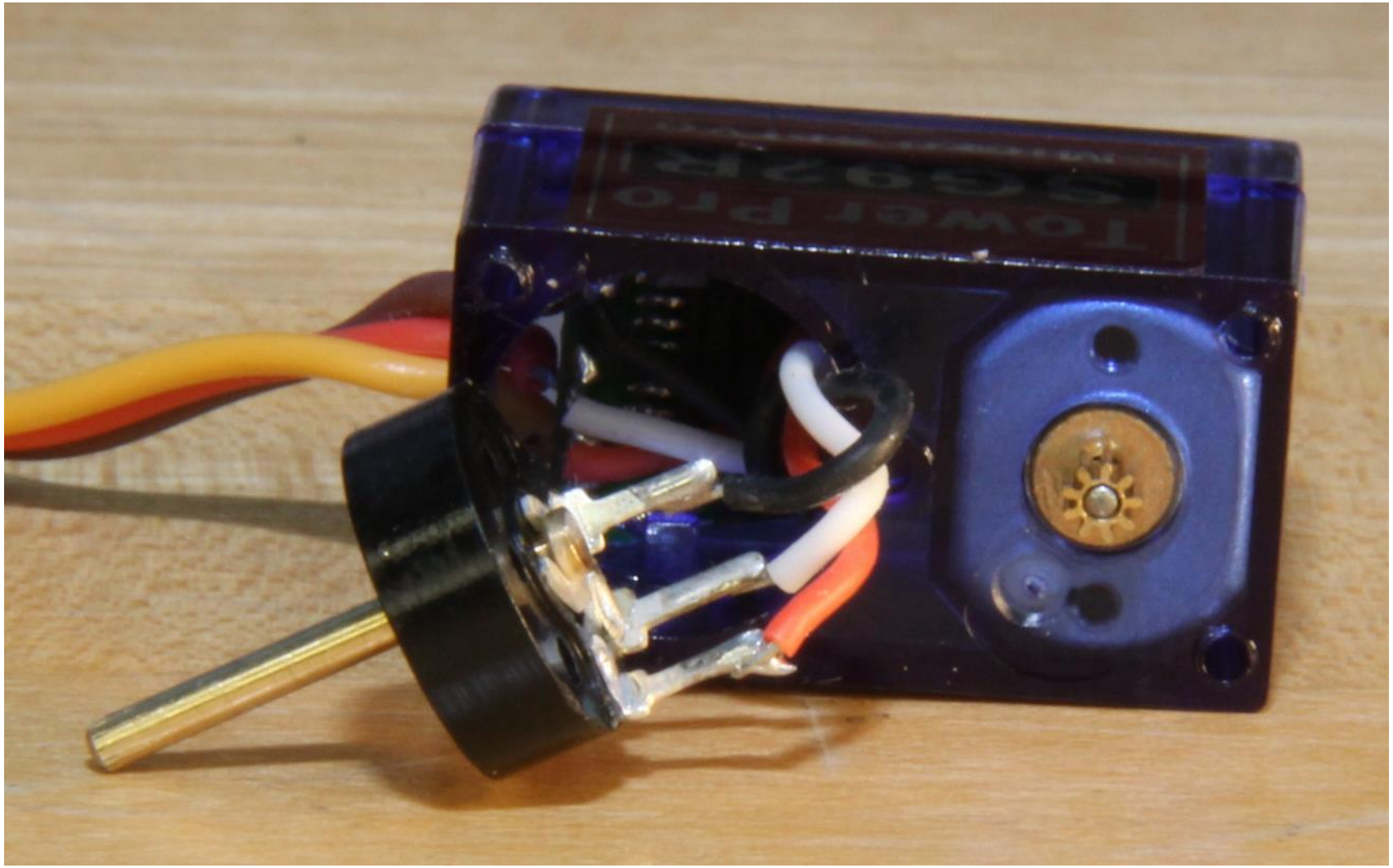


Internal Workings of Servo Drive

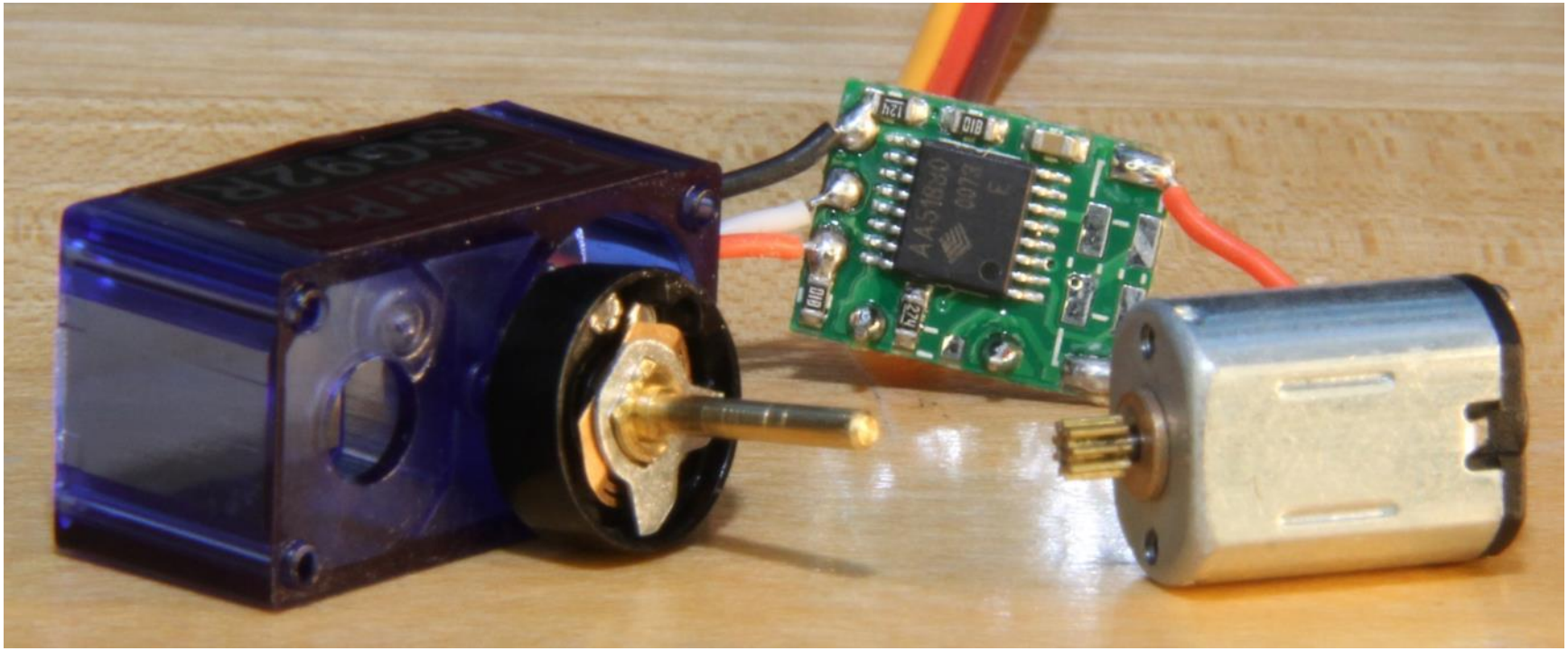
Servo Motor Gear Box



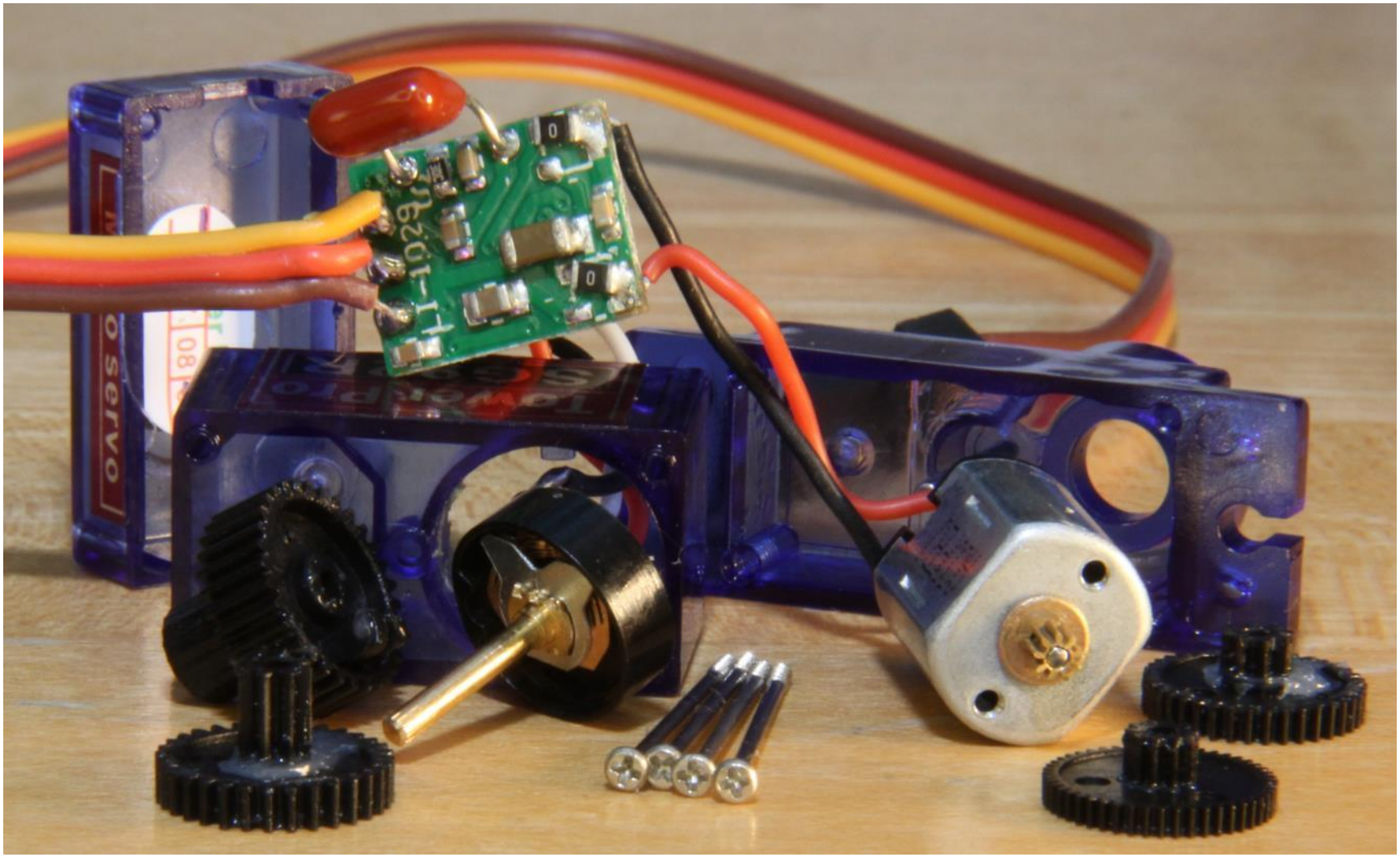
Servo Motor and Potentiometer



Servo Driver Board



Servo Motor Assembly



Servo Drive Construction

Rotates approximately 180 degrees (90° in each direction)

Can wire up with the standard Arduino Servo Library
(http://www.arduino.cc/en/Reference/Servo)

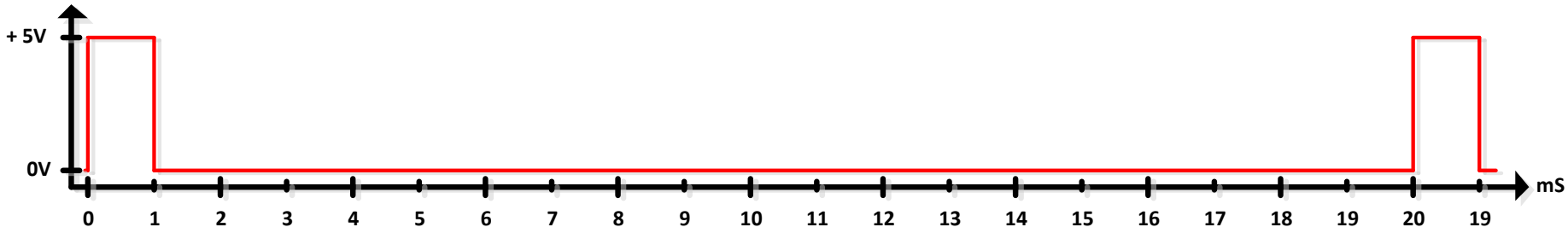
Orange Wire = Control	pin 9 or 10
Red Wire = 5VDC Power	pin
Brown Wire = Ground	pin

Position “0” = approximately 1.5ms pulse [middle of servo]

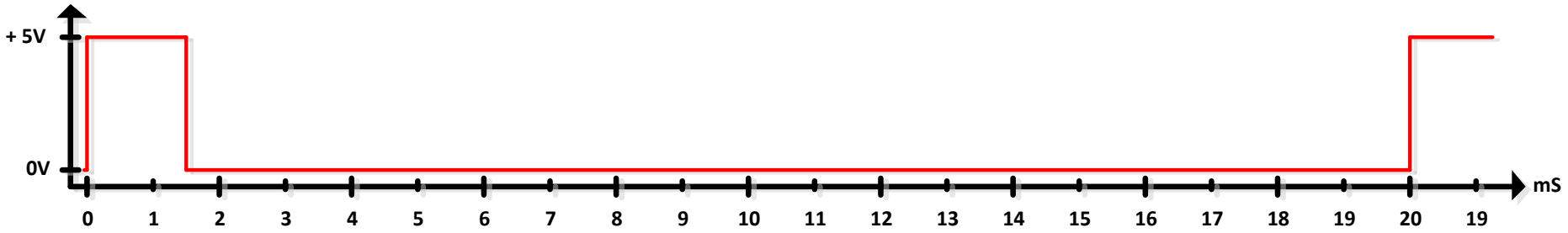
Position “left” = approximately 2ms pulse

Position “right” = approximately 1ms pulse

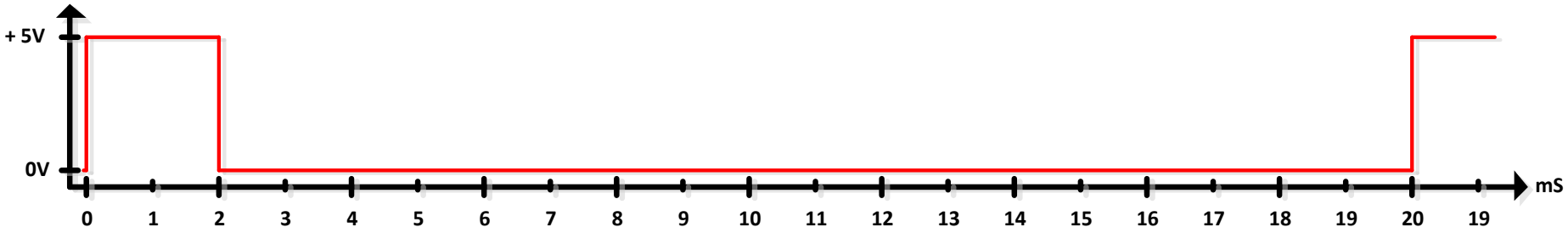
Servo Drive Right-most Position



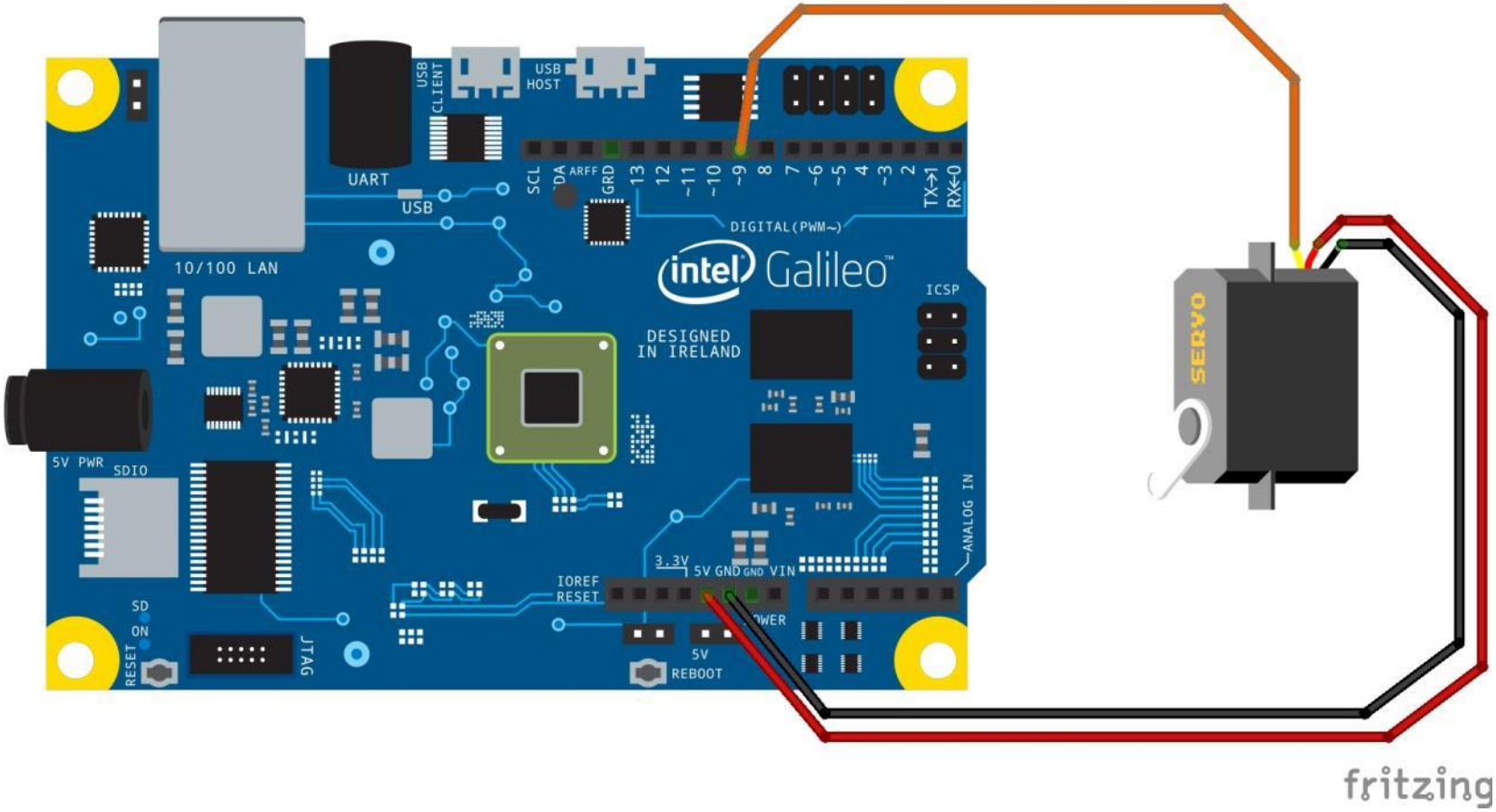
Servo Drive Center Position



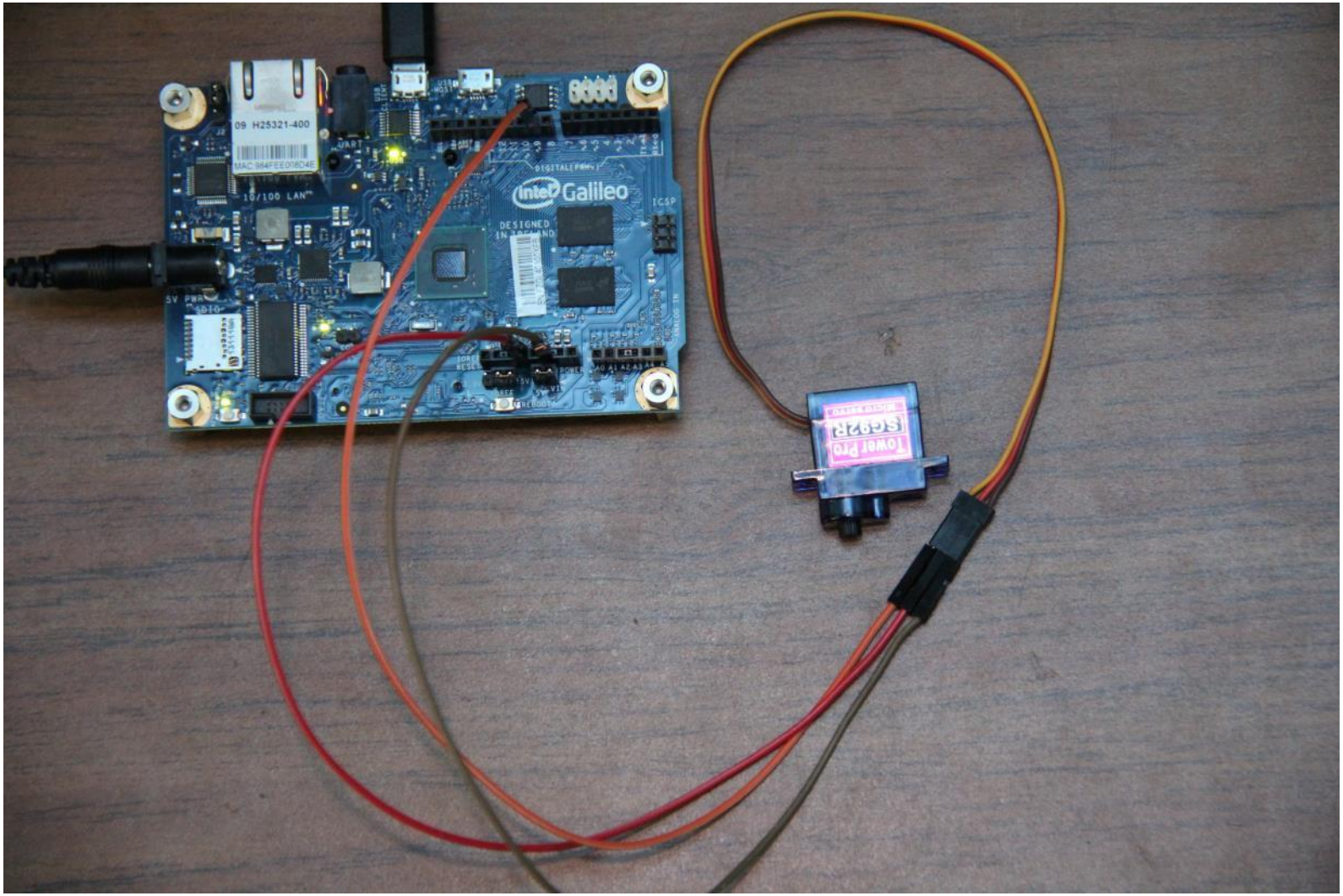
Servo Drive Left-most Position



Servo Motor Wire Diagram



Servo Motor Wiring



Servo Library

Declare a variable name for your servo

Syntax

Servo.attach(pin, min, max)

Parameter

"*Servo*" is the name you made for your servo, a variable

"pin" is the board pin number that you want to control your servo

"min" is the setting (in µsec) for a 0° angle (default = 544)

"max" is the setting (in µsec) for a 180° angle (default = 2400)

Note: "min" and "max" are optional parameters

Servo Library (continue)

To control the position of the servo

Syntax

Servo.write(angle)

Parameter

"*Servo*" is the name you made for your servo, a variable

"angle" the angle (in degrees) you want the servo to move to

from 0° to 180°, with 90° being the midpoint of the servo

Servo Library (continue)

Finite servo control… find the servo's limits

Syntax

Servo.writeMicroseconds(µS)

Parameters

"*Servo*" is the name you made for your servo, a variable

"µS" the "on" time pulse-width in microseconds

Script; Basic Servo Drive

/*

This is basic code to enable a small Servo. In this example we use the Micro Servo #169 from Adafruit and the "Servo Library".

The Micro Servo's red and black wires are for power. We tie the red wire to the Galileo board's +5 Volt power pin, and the black wire to the Galileo board's GND pin. The Micro Servo's orange wire is it's control wire, and we hook that up to one of the Galileo's Pulse-Width-Modulation (PWM) pins; pin 9.

We are driving the Micro Servo directly from the I/O pins on the Galileo.

NOTE: If driving multiple servos, provide an auxiliary +5 Volt power rail.

Script; Basic Servo Drive –continue-

A 1.0ms pulse width = all the way to the right.

A 1.5ms pulse width = middle position.

A 2.0ms pulse width = all the way to the left.

You may need to play with the minimum & maximum pulse widths for a particular servo. My current servo must run from 525ms to 3100ms to enable the full swing.

*/

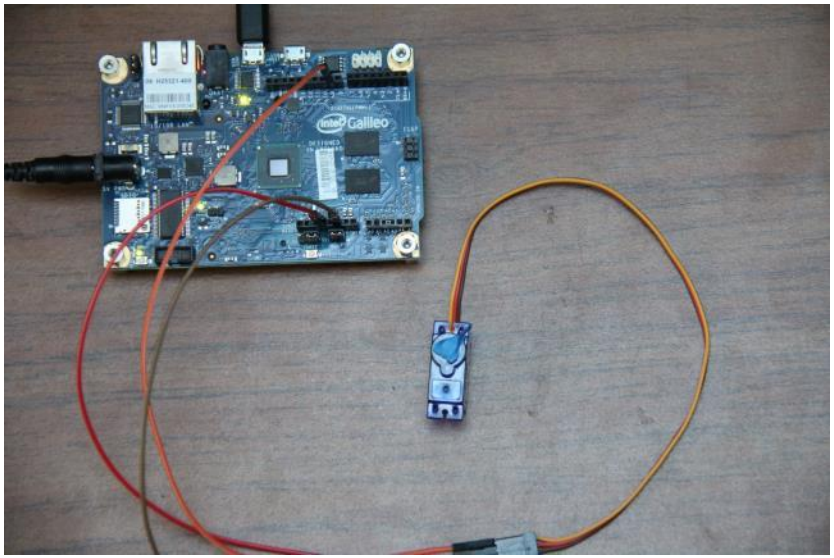
Script; Basic Servo Drive –continue-

```
#include <Servo.h>

Servo Servo_01;


void setup() {
  Servo_01.attach(9);              // use pin 9 to control the servo
  //Servo_01.write(90);            // to center the servo
}
```

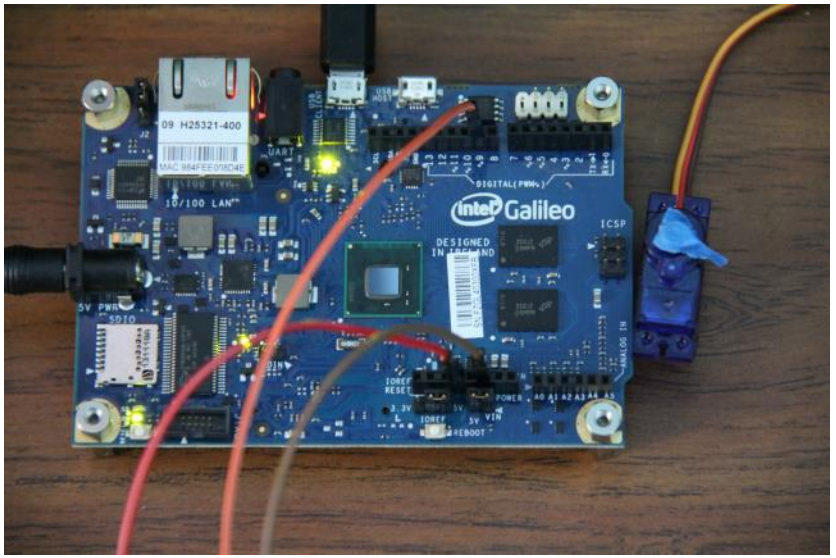
Script; Basic Servo Drive –continue-

```
void loop() {
  Servo_01.writeMicroseconds(1500);          // to center the servo
}
```

No two servos are quite the same

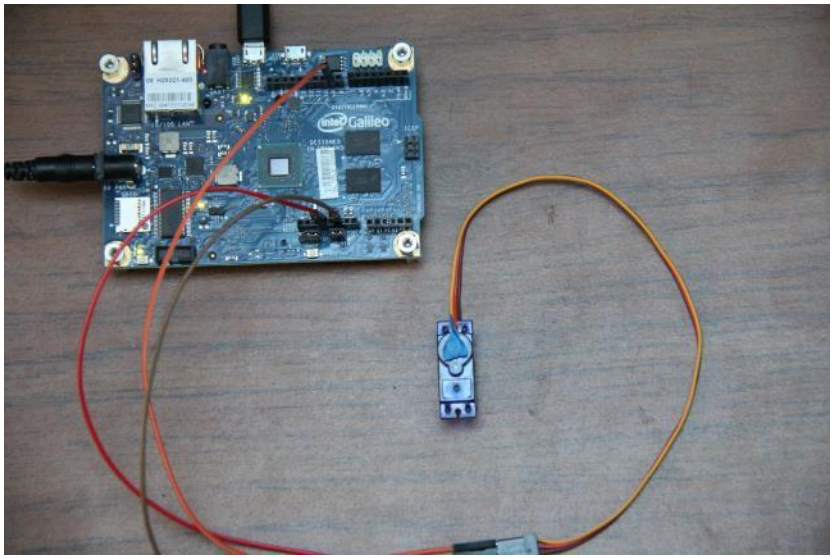


1.0ms Pulse Width

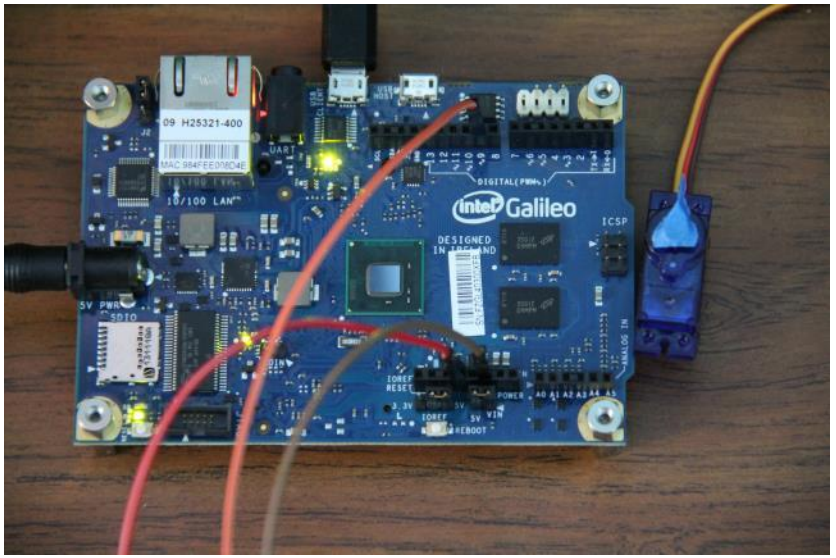


0.55ms Pulse Width

No two servos are quite the same

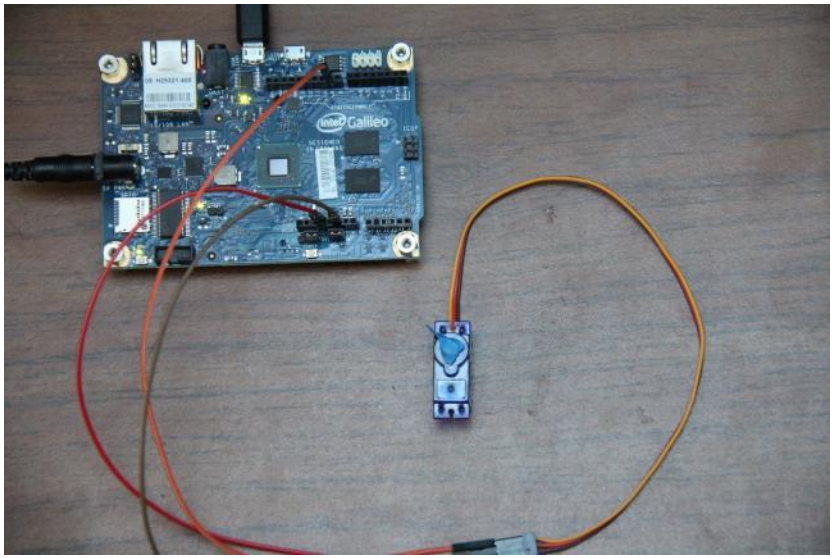


1.5ms Pulse Width

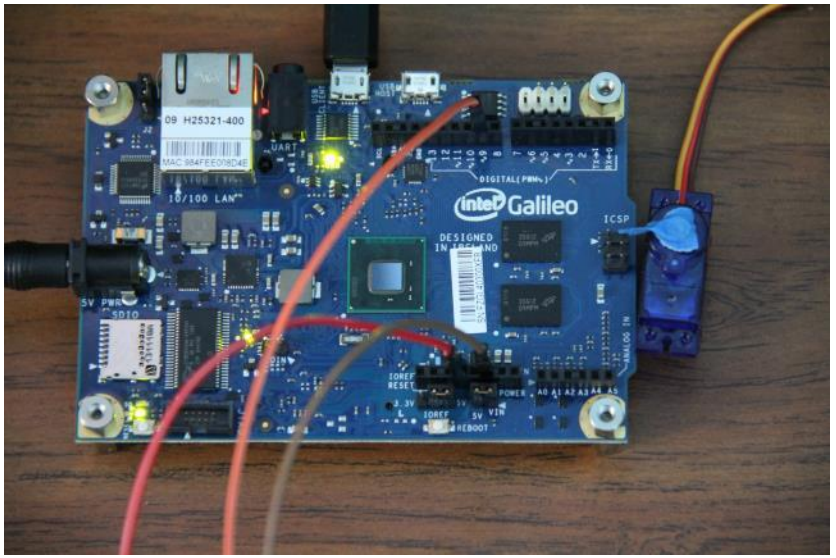


1.5ms Pulse Width

No two servos are quite the same



2.0ms Pulse Width



3.2ms Pulse Width


intel

Legal Disclaimer

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: http://www.intel.com/design/literature.htm%20 Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit Intel Performance Benchmark Limitations

All products, computer systems, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Celeron, Intel, Intel logo, Intel Core, Intel Inside, Intel Inside logo, Intel. Leap ahead., Intel. Leap ahead. logo, Intel NetBurst, Intel SpeedStep, Intel XScale, Itanium, Pentium, Pentium Inside, VTune, Xeon, and Xeon Inside are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Intel® Active Management Technology requires the platform to have an Intel® AMT-enabled chipset, network hardware and software, as well as connection with a power source and a corporate network connection. With regard to notebooks, Intel AMT may not be available or certain capabilities may be limited over a host OS-based VPN or when connecting wirelessly, on battery power, sleeping, hibernating or powered off. For more information, see http://www.intel.com/technology/iamt.

64-bit computing on Intel architecture requires a computer system with a processor, chipset, BIOS, operating system, device drivers and applications enabled for Intel® 64 architecture. Performance will vary depending on your hardware and software configurations. Consult with your system vendor for more information.

No computer system can provide absolute security under all conditions. Intel® Trusted Execution Technology is a security technology under development by Intel and requires for operation a computer system with Intel® Virtualization Technology, an Intel Trusted Execution Technology-enabled processor, chipset, BIOS, Authenticated Code Modules, and an Intel or other compatible measured virtual machine monitor. In addition, Intel Trusted Execution Technology requires the system to contain a TPMv1.2 as defined by the Trusted Computing Group and specific software for some uses. See http://www.intel.com/technology/security/ for more information.

†Hyper-Threading Technology (HT Technology) requires a computer system with an Intel® Pentium® 4 Processor supporting HT Technology and an HT Technology-enabled chipset, BIOS, and operating system. Performance will vary depending on the specific hardware and software you use. See www.intel.com/products/ht/hyperthreading_more.htm for more information including details on which processors support HT Technology.

Intel® Virtualization Technology requires a computer system with an enabled Intel® processor, BIOS, virtual machine monitor (VMM) and, for some uses, certain platform software enabled for it. Functionality, performance or other benefits will vary depending on hardware and software configurations and may require a BIOS update. Software applications may not be compatible with all operating systems. Please check with your application vendor.

* Other names and brands may be claimed as the property of others.

Other vendors are listed by Intel as a convenience to Intel's general customer base, but Intel does not make any representations or warranties whatsoever regarding quality, reliability, functionality, or compatibility of these devices. This list and/or these devices may be subject to change without notice.

DC Motor Technical Details

1 revolution every 1.5 seconds (24 rpm)

224:1 Gear Ratio

7mm double-flat Offset Shaft

48.6 in*oz. of torque at 3V

Built in clutch, limiting at 60 in*oz.

40mA run current, 400mA stall current

55mm x 48mm x 22.7mm

31.4 grams/1.11 ounces

#2x1/4” SMS screw

https://solarbotics.com/product/gm2/

Stepper Motor Technical Details

Bipolar Motor

7.5 degree stride angle

2-Phase

12VDC

400mA

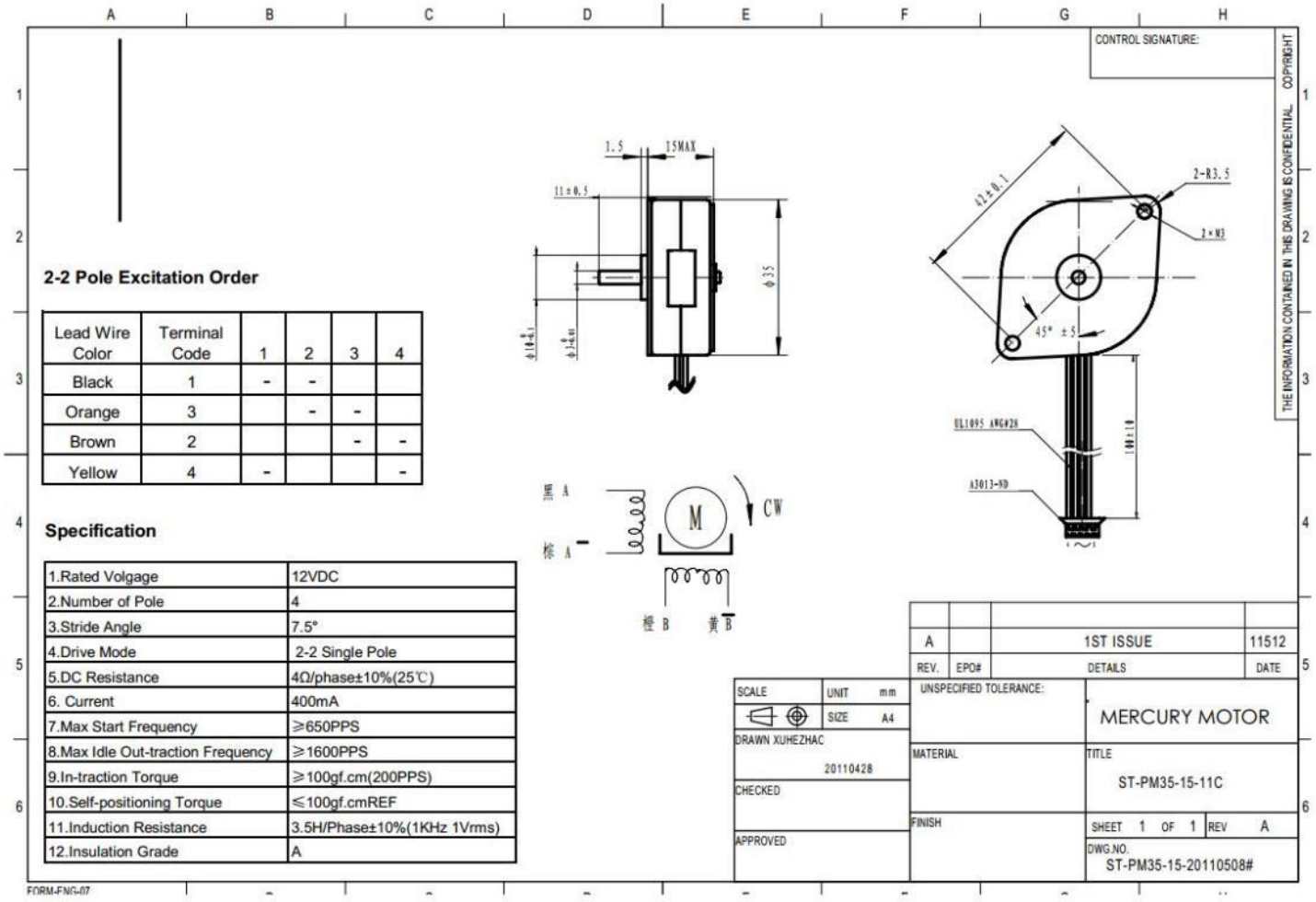
3mm diameter drive shaft

4-Wire Cable Attached

100 g/cm in-traction torque

https://www.sparkfun.com/products/10551

Stepper Motor Data Sheet



2-2 Pole Excitation Order

Lead Wire Color	Terminal Code	1	2	3	4
Black	1	-	-		
Orange	3		-	-	
Brown	2			-	-
Yellow	4	-			-

Specification

1.Rated Volgage	12VDC
2.Number of Pole	4
3.Stride Angle	7.5°
4.Drive Mode	2-2 Single Pole
5.DC Resistance	4Ω/phase±10%(25℃)
6. Current	400mA
7.Max Start Frequency	≥650PPS
8.Max Idle Out-traction Frequency	≥1600PPS
9.In-traction Torque	≥100gf.cm(200PPS)
10.Self-positioning Torque	≤100gf.cmREF
11.Induction Resistance	3.5H/Phase±10%(1KHz 1Vrms)
12.Insulation Grade	A

CONTROL SIGNATURE:

THE INFORMATION CONTAINED IN THIS DRAWING IS CONFIDENTIAL COPYRIGHT

REV.	EPO#	DETAILS	DATE
A		1ST ISSUE	11512

SCALE	UNIT mm
	SIZE A4
DRAWN XUHEZHAC 20110428	UNSPECIFIED TOLERANCE:
CHECKED	MATERIAL
APPROVED	FINISH

MERCURY MOTOR

TITLE: ST-PM35-15-11C

SHEET 1 OF 1 REV A

DWG.NO. ST-PM35-15-20110508#

FORM-ENG-07

Servo Technical Details

23 x 11 x 29 mm

3.0 to 6.0 VDC

9g or 0.32oz in weight

0.12 sec/60 degrees (at 4.8V)

1.6 kg-cm torque

-30° C to approximately +60° C operating temperature

Teflon bushing

19cm wire

Coreless motor

http://www.adafruit.com/product/169