11: Motors with Analog Inputs

Introduction:

In the previous lecture, different types of motors with digital interfaces to a Pi-Pico were presented. Other than the BLDC motor, these were fairly low power:

- BLDC: 6.0V @ 50A = 300W
- Stepper Motor: 5V @ 3A = 15W
- Digital Servo Motor: 5V @ 2A = 10W

If you need motor power, larger motors are needed. These typically have analog inputs.

This lecture looks at two types of motors and ways to interface these to a Pi-Pico:

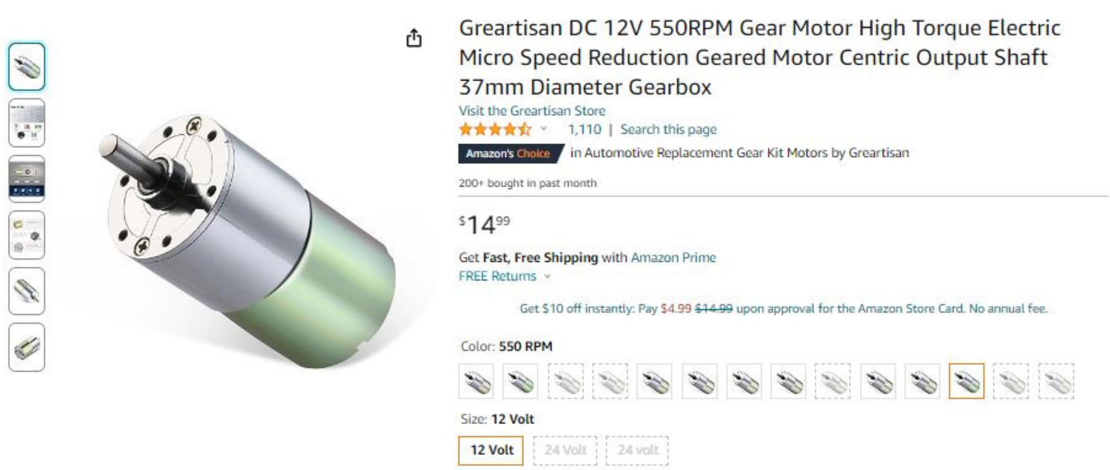
- DC Servo Motors
- 3-Phase AC Synchronous Motors

DC Servo Motors (6W)

A DC servo motor is the motor you're probably familiar with:

- If you apply a DC voltage to the motor, it spins.
- If you increase the DC voltage, it spins faster.

These are also the oldest type of electrical motor, dating back to 1900 when they were called *dynamos.*



12V, 6 Watt (500mA) DC Servo Motor

In terms of hardware, there are several options. If you need less than 6 Watts of power, a good choice is a Greartisan DC Gearhead Motor (below). This is a DC servo motor with a gear attached - allowing the no-load speed to vary from 5rpm to 600rpm.

In terms of the motor driver, what you need depends upon whether the motor just goes one direction or whether it can go forwards and in reverse.

Uni-Directional Hardware: If the motor just spins in one direction, a simple BJT transistor switch can be used to connect the Pi-Pico to the motor. Assuming a Zetex 1051A again, the base current needs to be:

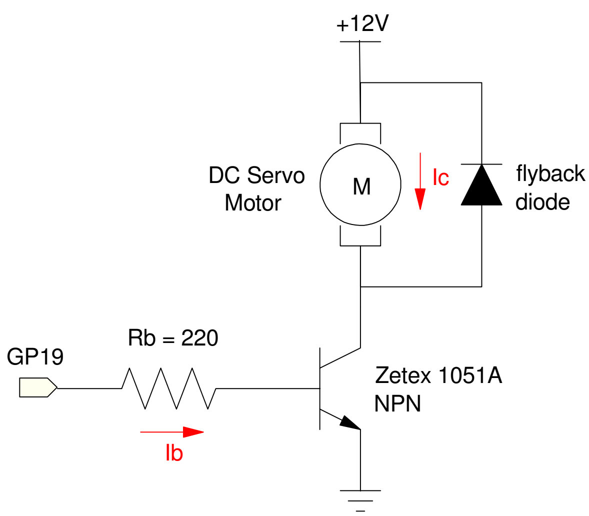
$$I_c(\max) = 500mA$$

$$h_{fe} \cdot I_b > I_c$$

$$12mA > I_b > \left(\frac{500mA}{300}\right) = 2.67mA$$

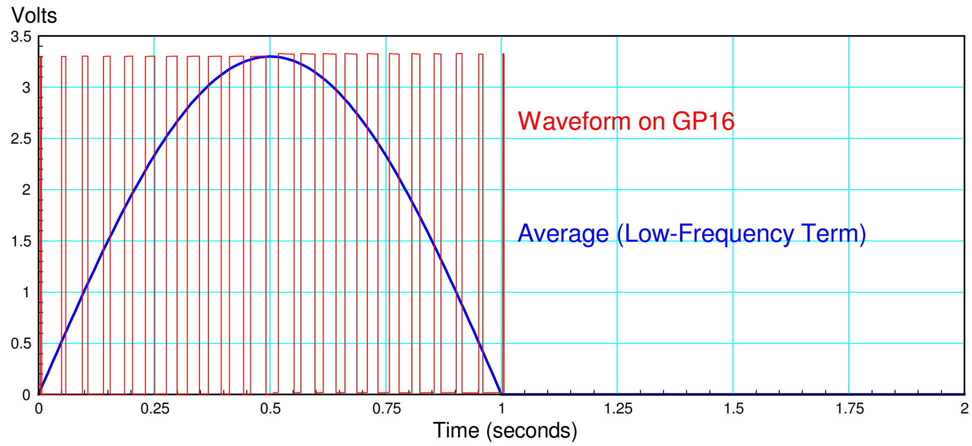
$$R_b = \left(\frac{3.3V - 0.7V}{I_b}\right)$$

$$216\Omega < R_b < 975\Omega$$



Hardware Connection for driving a DC motor in one direction
This configuration can be used for loads up to 3.5A

Pulse-width modulation can then be used to control the speed of the motor. For example, to make the motor speed up and slow down as follows:



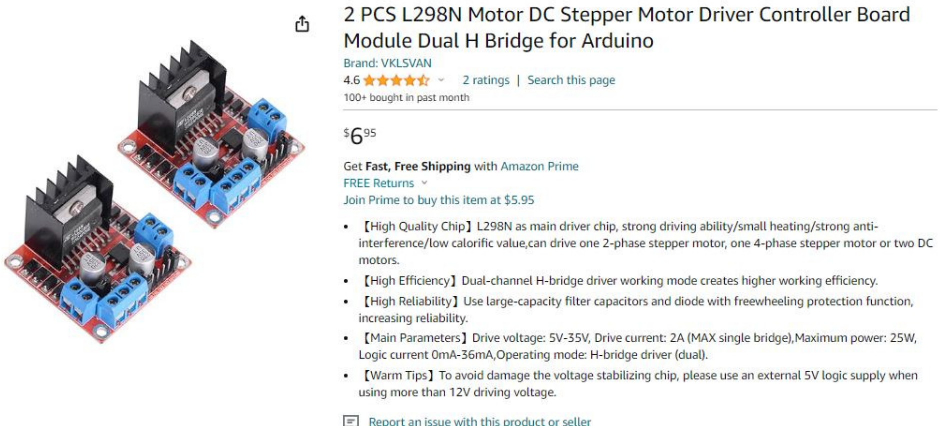
PWM used on GP16 to vary the speed of the DC motor

The following code can be used.

- Note: this is almost the same code used before. The only difference is increasing the PWM frequency to 20kHz. The frequency of the PWM produces an audible sound at the motor. Keeping this above 20kHz keeps this out of the audible range.

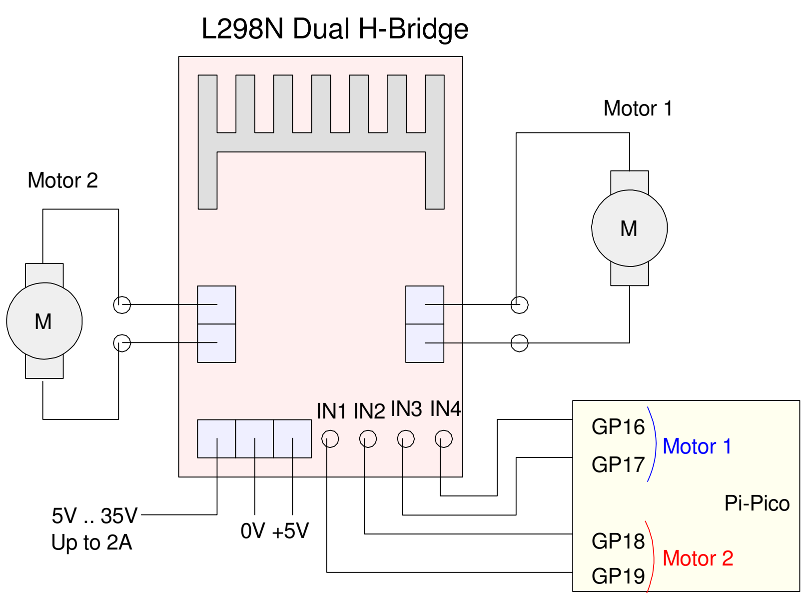
```
from machine import Pin, PWM
from time import sleep_ms
from math import sin, pi

Aout = Pin(16, Pin.OUT)
Aout = PWM(Pin(16))
Aout.freq (20_000)

Table = []
for i in range(0,100):
    Table.append(int(65535*sin(i*pi/100)))
for i in range(0,100):
    Table.append(0)

i = 0
while(1):
    i = (i + 1) % 200
    Aout.duty_u16(Table[i])
    sleep_ms(10)
```

Bidirectional Motion: If you need the motor to be able to spin both directions, then an H-bridge can be used along with PWM.



L298N dual H-bridge drivers can be used to drive the motor forward and reverse

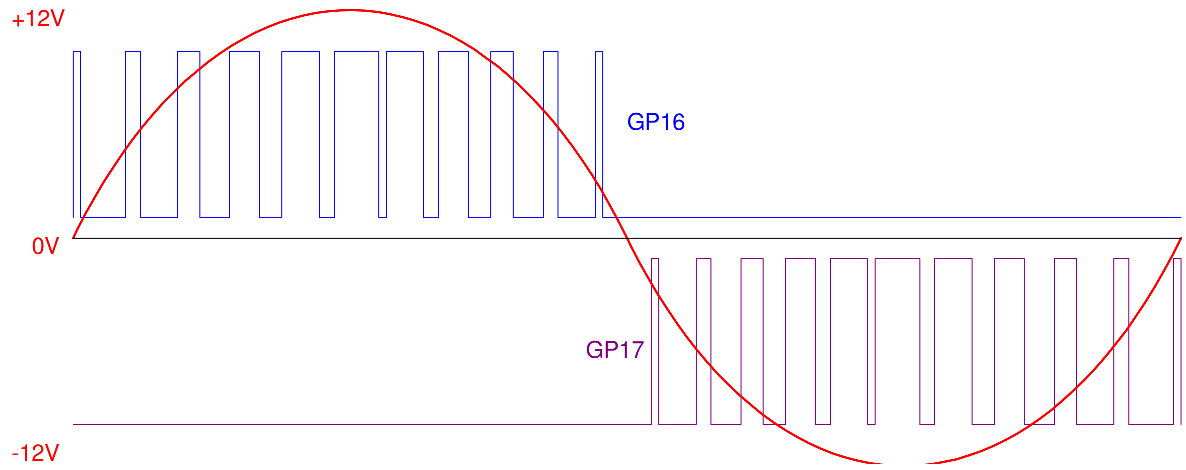
The hardware connection is similar to what was used for a stepper motor:



Two DC motors can be driven with a dual H-bridge driver

In terms of software, PWM again sets the speed of the motor with the direction set by which pin gets the PWM signal

	GP16	GP17
Forward	PWM	0V
Reverse	0V	PWM



The DC motor can be driven at 0..+12V using PWM on GP16 and 0V..-12V using PWM on GP17

The code is almost the same as before:

```
from machine import Pin, PWM
from time import sleep_ms
from math import sin, pi

Aout = Pin(16, Pin.OUT)
Aout = PWM(Pin(16))
Aout.freq (20_000)

Aout = Pin(17, Pin.OUT)
Aout = PWM(Pin(16))
Aout.freq (20_000

Table = []
for i in range(0,100):
    Table.append(int(65535*sin(i*pi/100)))
for i in range(0,100):
    Table.append(0)

i = 0
while(1):
    i = (i + 1) % 200
    if(Table[i] > 0):
        Aout.duty_u16(Table[i])
        Bout.duty_u16(0)
    else:
        Aout.duty_u16(0)
        Bout.duty_u16(-Table[i])
    sleep_ms(10)
```

DC Servo Motors (100W - 1000W)

If you need more power, use a different DC servo motor. The following for example is a motor for an

24V, 300 Watt DC motor (20A). Motor weight = 4 Lbs

In terms of hardware, everything remains the same except you need to increase the current capability to 20A.

For unidirectional motion, a power MOSFET can be used. Searching Digikey for

- MOSFET
- N-Channel
- 30A+
- Through Hole

results in 1,252 options. Selecting an IRFB7545PBF Mosfet

- Vds(max) = 60V
- Rds(on) = 5.9mOhm @ 57A @ 10V
- 4010pF @ 25V
- Vgs(th) (max) = 3.7V
- Ids(max) = 95A

What this tells you is:

- If you set Vgs < 3.7V, the MOSFET turns off (good)
- If you set Vgs = 10V, the on resistance is 5.9mOhm (good)
- At max current (20A), the Mosfet drops V = IR = 11.8mV and dissipates 236mW (good)
- This Mosfet can handle up to 95A (overkill)

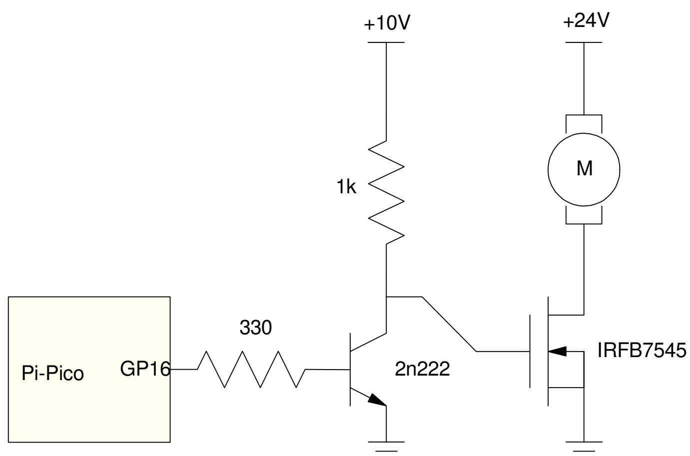
Mosfets will almost always out-perform BJT transistors.

IRFB7545PBF

DigiKey Part Number	448-IRFB7545PBF-ND
Manufacturer	Infineon Technologies
Manufacturer Product Number	IRFB7545PBF
Description	MOSFET N-CH 60V 95A TO220
Manufacturer Standard Lead Time	12 Weeks
Customer Reference	
Detailed Description	N-Channel 60 V 95A (Tc) 125W (Tc) Through Hole TO-220
Datasheet	Datasheet
EDA/CAD Models	IRFB7545PBF Models

One option for a MOSFET: An IRFB7545 is capable of 95A and 60V

A PIC can only output 3.3V. To bring this up to 10V, an NPN transistor can be used:

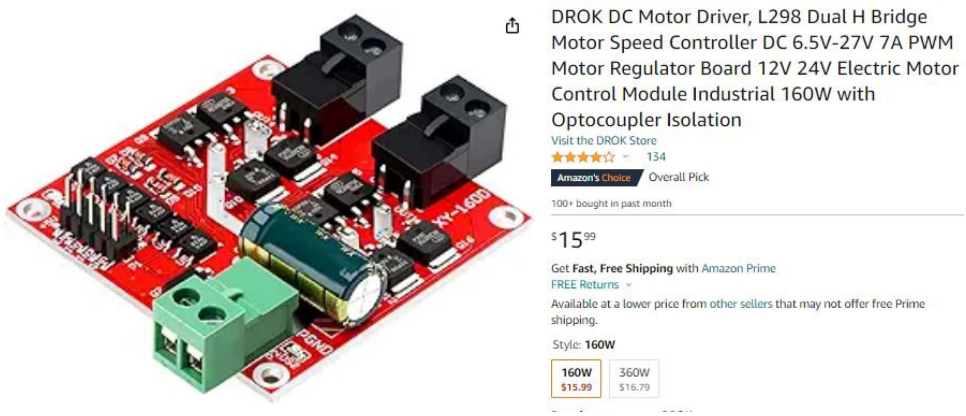


The capacitance of the Mosfet set the maximum switching frequency. The RC time constant is

$$t = RC = (1k\Omega)(4010pF) = 4.01\mu s$$

3 time constants is roughly how long it takes the Mosfet to turn on or off (121us). Inverting this circuit can run at up to 80kHz. Using 20kHz PWM should likewise be fine. (Same code as before).

For bi-directional motion, use an H-bridge capable of more current such as a DROK DC Motor Driver. Replace the previous H-bridge with this one and you're good to go for voltages up to 27V and currents up to 7A.

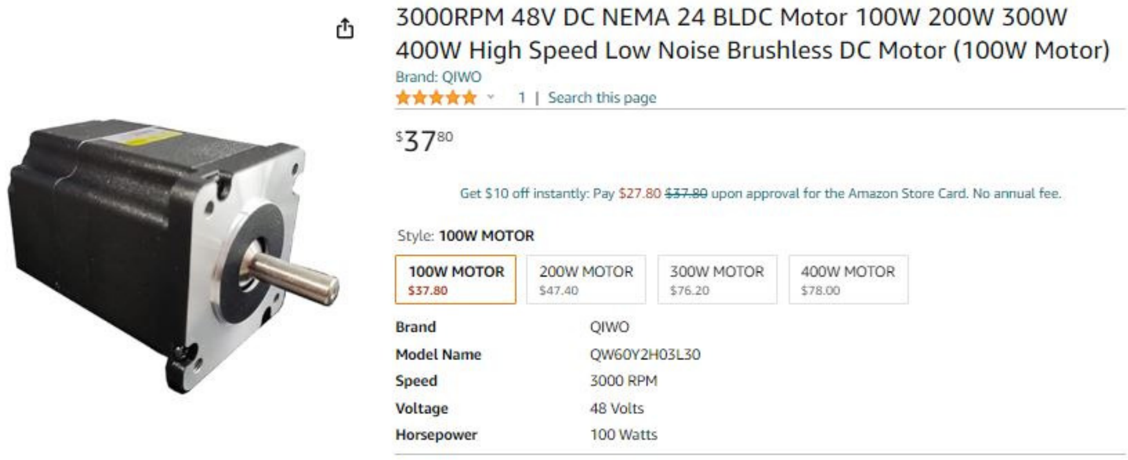


For even more power (up to 1200W)

- replace potentiometer with Pico output (voltage from PWM or D/A)
- replace SPDT switch with Pico output (voltage low or high for direction)

3-Phase AC Synchronous Motors (100W - 400W)

For larger motors, 3-Phase AC synchronous motors are usually the better option than DC motors. On Amazon, for example, you can buy a 400W motors for $78. To put this in perspective, a Tour-De-France athlete can output about 300W of power over long stretches and 1000W for a short burst. Put one of these motors on your bicycle and you can compete with world-class athletes.



3-Phase AC Synchronous Motor (BLDC motor)

Ever since about the year 2000, DC motors have been replaced by 3-phase AC synchronous motors, also known as *brushless DC motors* (BLDC). The reason is size, cost, efficiency and life.

- AC motors only have one set of coils for the rotor. DC motors, in contrast, have six or more sets of coils in the rotor. This reduces the ripple in the torque and voltage, but it also creates a motor that's 6x heavier than its AC counterpart.
- AC motors do not need the commutatiors that DC motors need - so there's less parts to wear out
- AC motors are inherently more efficient. With DC motors, when you switch out a coil in the rotor, all of the energy stored in the rotor's inductance $\left(\frac{1}{2}LI^2\right)$ is lost.
- AC motors produce less RF interference. When the coils of a DC motor are switched out, a spark is created as the magnetic field collapses and the energy stored is dissipated. These sparks create RF interference, requiring greater shielding for DC motors.

The problem with 3-phase AC motors is the motor's input needs to be a 3-phase sine wave where the frequency sets the speed. Fortunately, driver boards can be purchases on Amazon for about $20. For this board:

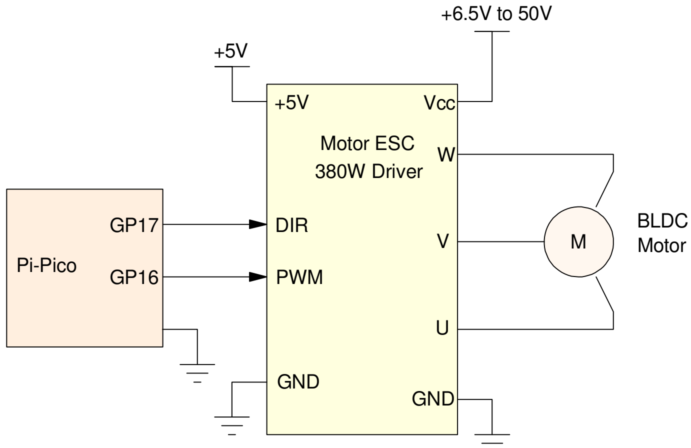
- The PWM signal sets the speed of the motor from 0% to 100%
- The Direction Control pin sets the direction of the motor (CW or CCW)
- Maximum speed is 74,000 rpm with a 6-pole motor

BLDC Motor Driver with PWM input.

Connections to a Pi-Pico board use two pins:

- One to set the speed (PWM),
- The other to set the direction (DIR)

Both 3.3V and 5V logic levels work for the input. The board itself needs 5V to operate its electronics.



300W Brushless Motor connections

There are variations of these controllers which take analog inputs (0..3.3V). PWM signals are much easier to create, however, so the trend seems to be going towards PWM inputs for everything.