# Design of an electrical actuator system

Design of an Electrical Actuator Operating System for Two Actuators with Linear Extension and Retraction in Synchronization using a Microcontroller.

## Problem Definition

**1.** Using a suitable microcontroller:

- Design an Electrical Actuator Extension and Retraction System operated by a microcontroller.
- Choose an electrical actuator commercially available that extends 15 cm linearly and retracts 10 cm out of the 15 cm total extension.
- Select practically available components, understand their parameters, and incorporate them into the design.
- Choose a practical power supply (battery) and calculate the maximum power consumption during the extension and retraction.

**2.** Create an actual hardware-based design using commercially available components and/or kits.

## Requirements

1. Linear Extension: 15 cm
2. Linear Retraction: 10 cm
3. Total Extension Capability: 15 cm (out of which 10 cm is retraction)

## Calculations

Assume the lead screw pitch (linear displacement per revolution) of the actuator is "P" cm/rotation.

1. Number of rotations for Linear Extension (15 cm): Total number of rotations for 15 cm extension = (Total extension length) / (Lead screw pitch) Number of rotations for extension = 15 cm / P
2. Number of rotations for Linear Retraction (10 cm): Total number of rotations for 10 cm retraction = (Total retraction length) / (Lead screw pitch) Number of rotations for retraction = 10 cm / P

# Components required

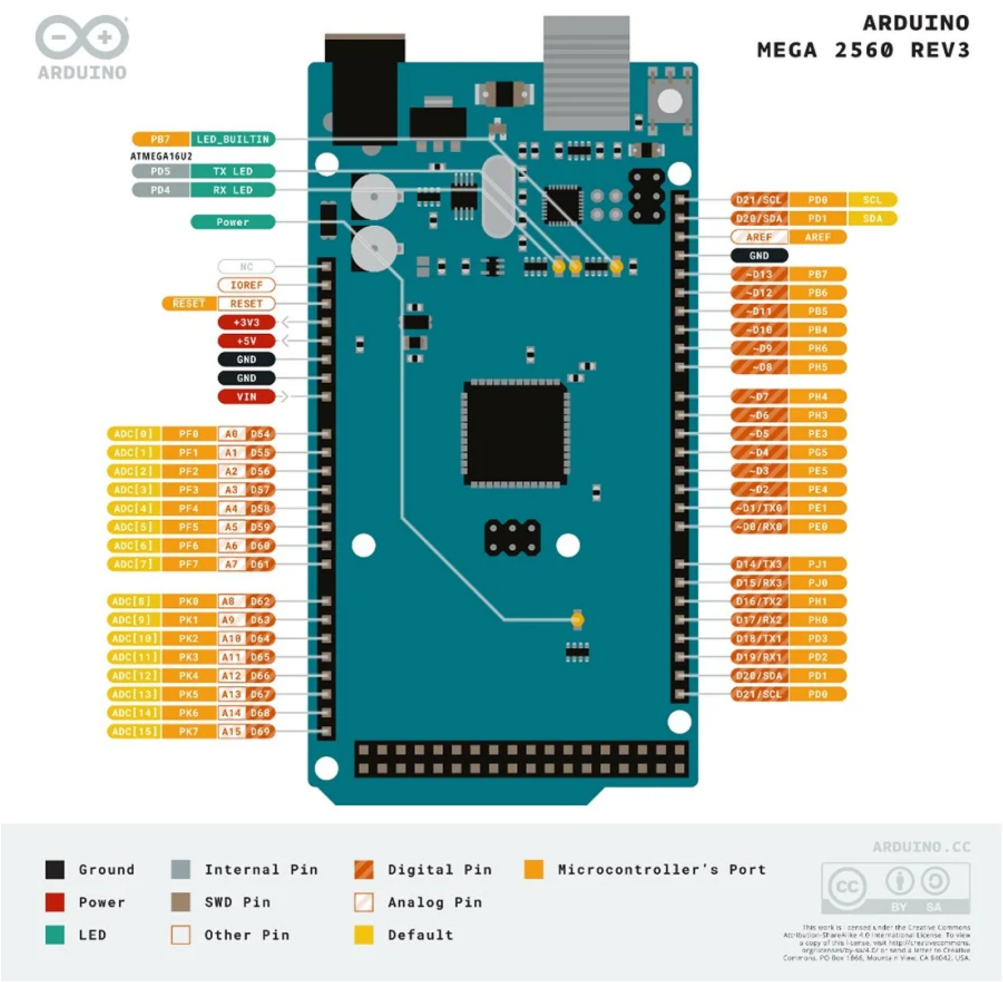
- 12VDC Power Supply
- Arduino Mega / Arduino UNO
- LCD with Buttons
- 2 Channel Relay (LN298)
- Actuator (12VDC with max. 10A current draw)
- USB Cable Type A/B Jumper Wires

# Algorithm for Electrical Actuator Extension and Retraction

1. Define the necessary constants and pins:
    - Set a constant for the lead screw pitch of the actuator in cm/rotation.
    - Define the digital pin connected to the actuator motor control.
2. Setup:
    - Set the actuator control pin as an output.
3. Loop:
    - Calculate the number of rotations required for the actuator's linear extension based on the total extension distance and the lead screw pitch.
    - Use a loop to execute the following steps for the calculated number of rotations:
        - Apply the appropriate polarity to the actuator to extend it.
        - Add a delay to allow the actuator to move (adjust the time as needed).
        - Turn off the actuator.
        - Add a delay before the next rotation (adjust the time as needed).
    - Calculate the number of rotations required for the actuator's linear retraction based on the total retraction distance and the lead screw pitch.
    - Use a loop to execute the following steps for the calculated number of rotations:
        - Apply the opposite polarity to the actuator to retract it.
        - Add a delay to allow the actuator to move (adjust the time as needed).
        - Turn off the actuator.
        - Add a delay before the next rotation (adjust the time as needed).
    - Add a delay before restarting the loop (adjust the time as needed).

# Wiring Connections using Arduino MEGA 2560

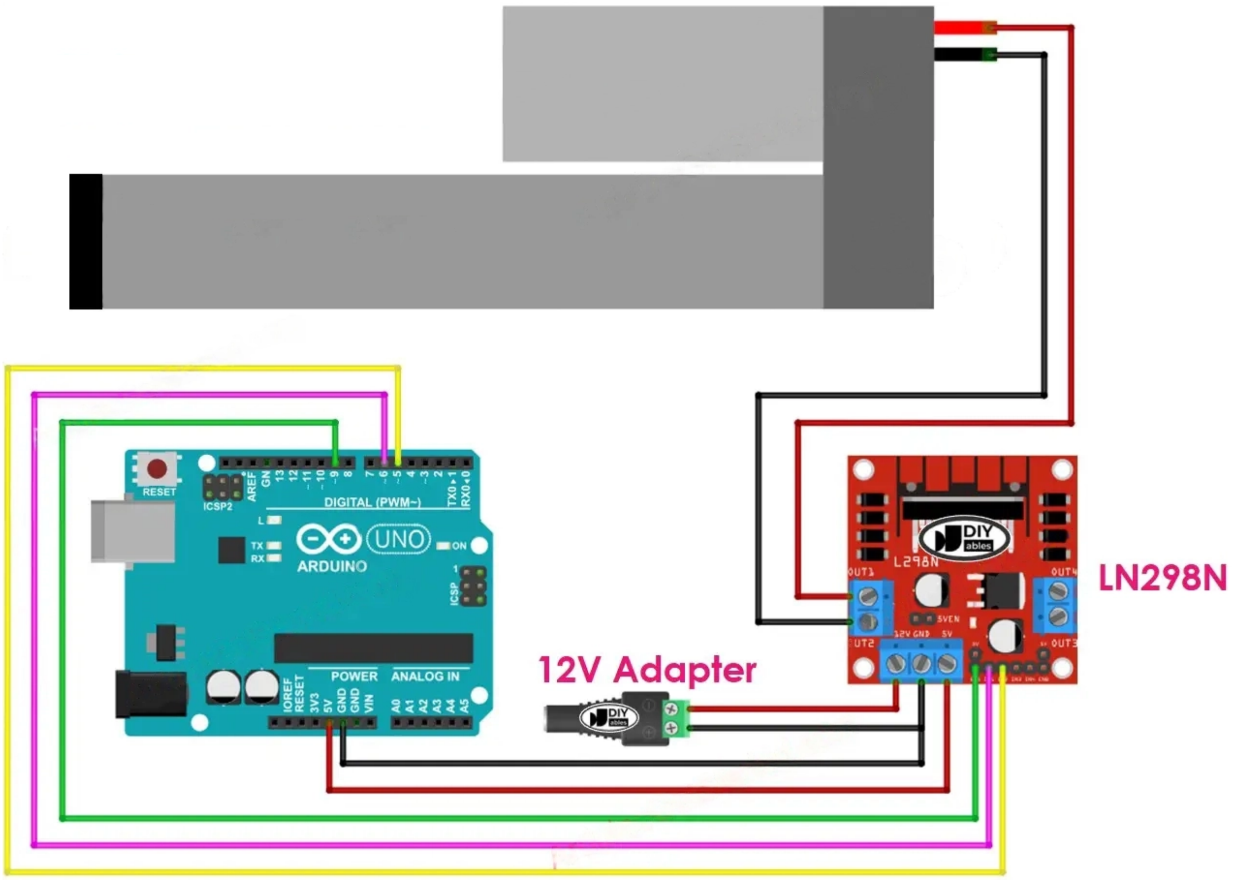
- LCD stacked on Arduino Pin 26
- Relay IN1 to Arduino Pin 30
- Relay IN2 to Arduino 5V
- Relay VCC to Arduino GND
- Relay GND to Relay NO2
- 12VDC to Relay NC2
- 12VDC to Relay NC1
- Relay NC2 to Relay NO1
- Relay NO2 to Actuator Positive
- Relay COM1 to Actuator Negative
- Relay COM2



# CODE

```
// Code by SUHANA ARSH, SADIQ ALI, VEERABHADRAYYA C ROOGI
#include <LiquidCrystal.h>
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
//pin assignment
int relay_1 = 26;  //relay 1 pin to activate coil
int relay_2 = 30;  //relay 2 pin to activate coil
int lcd_key = 0;   //LCD button press readings on Pin 0
const int btnup = 100; //value of sensor when up button is pressed
const int btndown = 255; //value of sensor when down button is pressed
const int threshold = 5;
void setup() {
  Serial.begin(9600);
  pinMode(relay_1, OUTPUT);
  pinMode(relay_2, OUTPUT);
  digitalWrite(relay_1, LOW);
  digitalWrite(relay_2, LOW);
  //LCD
  lcd.begin(16, 2);
  lcd.setCursor(0, 0);
  lcd.write("Press UP to ext");
  lcd.setCursor(0, 1);
  lcd.write("Press DN to ret");
}
void loop() {
  lcd_key = analogRead(A0); //reads if there is any input from button presses
  if (lcd_key > btnup - threshold && lcd_key < btnup + threshold) {
    digitalWrite(relay_1, HIGH);
    digitalWrite(relay_2, LOW);
  }
  else if (lcd_key > btndown - threshold && lcd_key < btndown + threshold) {
    digitalWrite(relay_1, LOW);
    digitalWrite(relay_2, HIGH);
  }
  else{
  }
}
```

# Wiring Connections using Arduino uno



# CODE

```
// Code by SUHANA ARSH, SADIQ ALI, VEERABHADRAYYA C ROOGI
void setup() {
pinMode(7, OUTPUT); // Configure pin 7 as an Output
pinMode(8, OUTPUT); // Configure pin 8 as an Output

digitalWrite(7, HIGH); // Initialize pin 7 as Low
digitalWrite(8, HIGH); // Initialize pin 7 as Low

}

void loop() {
  // Extend Linear Actuator
  digitalWrite(7, LOW);
  digitalWrite(8, HIGH);

  delay(2000); // 2 seconds

  // Stops Actuator
  digitalWrite(7, HIGH);
  digitalWrite(8, HIGH);

  delay(2000); // 2 seconds

  // Retracts Linear Actuator
  digitalWrite(7, HIGH);
  digitalWrite(8, LOW);

  delay(2000); // 2 seconds

  // Stop Actuator
  digitalWrite(7, HIGH);
  digitalWrite(8, HIGH);

  delay(2000); // 2 seconds
}
```

LINKS

https://github.com/sadiqalimir/actuatorOS.git

https://lastminuteengineers.com/two-channel-relay-module-arduino-tutorial/

https://arduinogetstarted.com/tutorials/arduino-actuator/

https://youtu.be/qlKOU57Vi2M