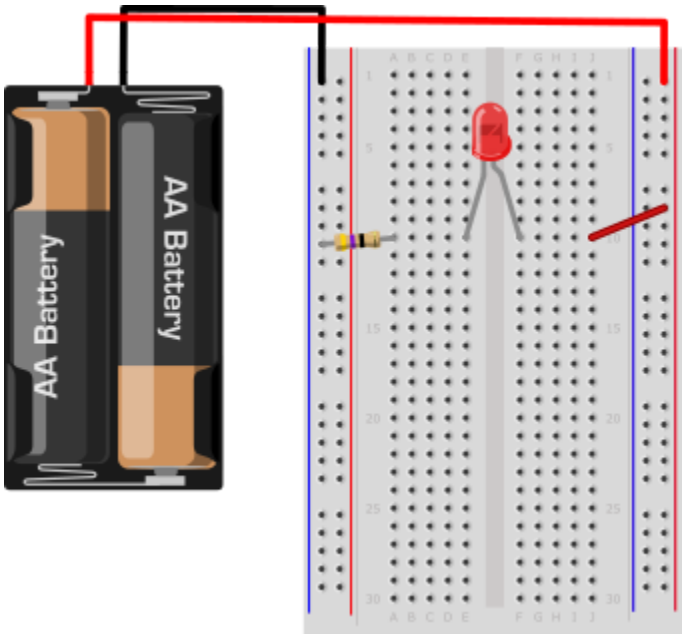
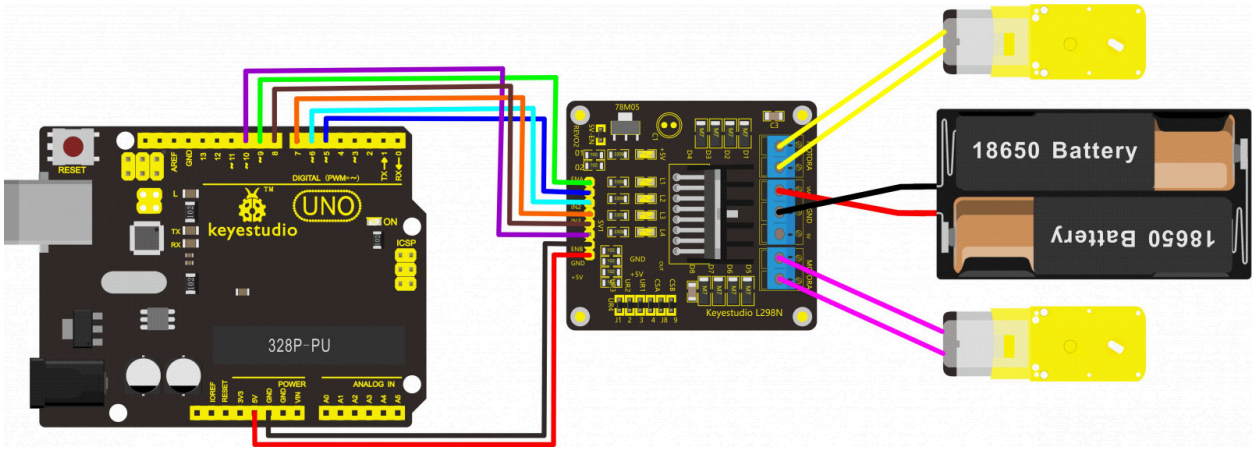
Information about the mechanics





- **Stepper Motors and Arduino - The Ultimate Guide**
  - https://www.youtube.com/watch?v=7spK_BkMJys
- **Coordinated stepper motor control (arduino)**
  - https://www.youtube.com/watch?v=fHAO7SW-SZI
  - https://www.iforce2d.net/sketches/

- **Kinetic Sculpture - Creative code - Coordinated stepper motor control**
    - https://www.youtube.com/watch?v=wiF0mZvuSRw
- **Control a NEMA 17 Stepper Motor with A4988 Driver and Arduino - Full Guide**
- https://www.youtube.com/watch?v=wcLeXXATCR4
-

https://docs.arduino.cc/learn/electronics/stepper-motors/

- **StepperOneRevolution**
- The motor should revolve one revolution in one direction, then one revolution in the other direction.

```
#include <Stepper.h>

const int stepsPerRevolution = 200;  // change this to fit the number of steps per revolution
// for your motor

// initialize the stepper library on pins 8 through 11:
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);

void setup() {
  // set the speed at 60 rpm:
  myStepper.setSpeed(60);
  // initialize the serial port:
  Serial.begin(9600);
}

void loop() {
  // step one revolution  in one direction:
  Serial.println("clockwise");
  myStepper.step(stepsPerRevolution);
```

```
  delay(500);

  // step one revolution in the other direction:
  Serial.println("counterclockwise");
  myStepper.step(-stepsPerRevolution);
  delay(500);
}
```

- Spin one direction, then the other for 1 revolution

```
#define DIR_PIN 2
#define STEP_PIN 3

int stepsPerRev = 200;       // how many steps = 1 full rotation (common value)
int stepDelay = 800;         // speed: delay between steps in microseconds

void setup() {
  pinMode(DIR_PIN, OUTPUT);
  pinMode(STEP_PIN, OUTPUT);
}

void loop() {
  // Forward 1 revolution
  digitalWrite(DIR_PIN, HIGH);
  for (int i = 0; i < stepsPerRev; i++) {
    digitalWrite(STEP_PIN, HIGH);
    delayMicroseconds(stepDelay);
    digitalWrite(STEP_PIN, LOW);
    delayMicroseconds(stepDelay);
  }
  delay(1000);

  // Backward 1 revolution
  digitalWrite(DIR_PIN, LOW);
  for (int i = 0; i < stepsPerRev; i++) {
    digitalWrite(STEP_PIN, HIGH);
    delayMicroseconds(stepDelay);
    digitalWrite(STEP_PIN, LOW);
    delayMicroseconds(stepDelay);
  }
  delay(1000);
}
```

Single repeating raindrop (with acceleration)

```
#define DIR_PIN    2
#define STEP_PIN   3
#define ENABLE_PIN 4

const int stepsPerRev = 200;   // adjust if your stepper is different

// --- Acceleration parameters ---
const int lowSpeed  = 2000;   // starting delay (slow)
const int highSpeed = 100;    // fastest delay
const int change    = 1;      // acceleration increment
const int accelRevs = 10;     // number of revs to accelerate over

void setup() {
 pinMode(DIR_PIN, OUTPUT);
 pinMode(STEP_PIN, OUTPUT);
 pinMode(ENABLE_PIN, OUTPUT);

 digitalWrite(ENABLE_PIN, LOW);  // enable driver
}

void simpleAccelOnce() {
 long totalSteps = (long)stepsPerRev * accelRevs;
 int d = lowSpeed;

 for (long i = 0; i < totalSteps; i++) {
   digitalWrite(STEP_PIN, HIGH);
   delayMicroseconds(2);       // minimum STEP pulse width
   digitalWrite(STEP_PIN, LOW);
   delayMicroseconds(d);

   if (d > highSpeed) {
     d -= change;
     if (d < highSpeed) d = highSpeed;
   }
 }
}

void stepMotor(long steps, int stepDelay) {
 for (long i = 0; i < steps; i++) {
```

```
   digitalWrite(STEP_PIN, HIGH);
   delayMicroseconds(2);
   digitalWrite(STEP_PIN, LOW);
   delayMicroseconds(stepDelay);
  }
}

void loop() {
 // --- Quick ramp-up ---
 digitalWrite(DIR_PIN, LOW);     // direction for ramp-up
 simpleAccelOnce();

 delay(100);                     // pause 1 second

 // --- Slow sustained move in opposite direction ---
 digitalWrite(DIR_PIN, HIGH);    // reverse direction
 stepMotor((long)stepsPerRev * 10, 1200);   // 10 revs, slow speed

 delay(2000);                     // pause 2 seconds

 // optional: repeat forever
 // loop will naturally repeat, so this pattern will continue
}
```

Raindrop (working) with the scheduling. Timed 3 times

```
// ================== CONFIG ==================
#define ALWAYS_HOLD 1          // keep driver enabled so it holds between runs
#define USE_DRIVER_RESET 1      // set to 1 if RESET/SLEEP are wired to Arduino; else 0

// Pins
#define DIR_PIN      2
#define STEP_PIN     3
#define ENABLE_PIN   4

#if USE_DRIVER_RESET
  #define RESET_PIN  8         // A4988 nRESET (active LOW)
  #define SLEEP_PIN  9         // A4988 nSLEEP (active LOW)
#endif

// Motor steps per rev (= 200 * microstep if using microstepping)
```

```
const int stepsPerRev = 200;

// Pulse width
const int STEP_HIGH_US = 2;

// Downward (drop) ramp
const int DOWN_START_DELAY_US = 2000;
const int DOWN_MIN_DELAY_US   = 400;
const int DOWN_CHANGE_US      = 1;
const int DOWN_REVS           = 5;   // CHANGED: was 10 → half the travel

// Upward (wind) ramp — conservative for torque
const int UP_START_DELAY_US = 3000;
const int UP_MIN_DELAY_US   = 1300;
const int UP_CHANGE_US      = 1;
const int UP_REVS           = 5;     // CHANGED: was 10 → half the travel

// Pre-tension (seats the line; net zero motion)
const int  PRETENSION_STEPS = 50;
const int  PRETENSION_US    = 2200;

// === Timeline in MILLISECONDS ===
const unsigned long scheduleMs[] = { 5000UL, 20000UL, 25000UL };
const size_t NUM_EVENTS = sizeof(scheduleMs) / sizeof(scheduleMs[0]);

// ================== GLOBALS ==================
unsigned long timelineStart = 0;
size_t nextEventIdx = 0;

// ================== HELPERS ==================
inline void holdEnable(bool on) { digitalWrite(ENABLE_PIN, on ? LOW : HIGH); }

inline void stepOnceDelay(int delayAfterLowUs) {
  digitalWrite(STEP_PIN, HIGH);
  delayMicroseconds(STEP_HIGH_US);
  digitalWrite(STEP_PIN, LOW);
  delayMicroseconds(delayAfterLowUs);
}

void stepN(long steps, int delayUs, bool dirHigh) {
  digitalWrite(DIR_PIN, dirHigh ? HIGH : LOW);
  delayMicroseconds(5);
  for (long i = 0; i < steps; i++) stepOnceDelay(delayUs);
}

void rampMove(long revCount, int startDelayUs, int minDelayUs, int changePerStepUs, bool
dirHigh) {
  digitalWrite(DIR_PIN, dirHigh ? HIGH : LOW);
  delayMicroseconds(5);
  long totalSteps = (long)stepsPerRev * revCount;
```

```
  int d = startDelayUs;
  for (long i = 0; i < totalSteps; i++) {
    stepOnceDelay(d);
    if (d > minDelayUs) {
      d -= changePerStepUs;
      if (d < minDelayUs) d = minDelayUs;
    }
  }
}

void pretension() {
  stepN(PRETENSION_STEPS, PRETENSION_US, /*down*/false);
  stepN(PRETENSION_STEPS, PRETENSION_US, /*up*/true);
}

void runSequenceOnce() {
  holdEnable(true);
  // DOWN / drop
  rampMove(DOWN_REVS, DOWN_START_DELAY_US, DOWN_MIN_DELAY_US,
DOWN_CHANGE_US, /*down*/false);
  delay(150);
  // UP / wind
  rampMove(UP_REVS, UP_START_DELAY_US, UP_MIN_DELAY_US, UP_CHANGE_US,
/*up*/true);
  delay(2000);
}

void driverInitDeterministic() {
  holdEnable(false);
  digitalWrite(DIR_PIN, LOW);
  digitalWrite(STEP_PIN, LOW);

#if USE_DRIVER_RESET
  pinMode(RESET_PIN, OUTPUT);
  pinMode(SLEEP_PIN, OUTPUT);
  digitalWrite(SLEEP_PIN, HIGH); delay(2);
  digitalWrite(RESET_PIN, LOW);  delay(2);
  digitalWrite(RESET_PIN, HIGH); delay(2);
#else
  delay(100);
#endif

  holdEnable(true);
  delay(20);
  pretension();
  holdEnable(ALWAYS_HOLD);
}

// ================== ARDUINO ==================
void setup() {
```

```
  pinMode(DIR_PIN, OUTPUT);
  pinMode(STEP_PIN, OUTPUT);
  pinMode(ENABLE_PIN, OUTPUT);
  driverInitDeterministic();
  timelineStart = millis();
}

void loop() {
  if (nextEventIdx >= NUM_EVENTS) {
    if (!ALWAYS_HOLD) holdEnable(false);
    return;
  }
  unsigned long elapsed = millis() - timelineStart;
  if (elapsed >= scheduleMs[nextEventIdx]) {
    runSequenceOnce();
    nextEventIdx++;
  }
}
```

Test 1 for raindrop timing 28

drop	Time stamp in milli	Adjusted time for code in milli (-1500)
1	4560	3060
2	13647	12147
3	20004	18504
4	28130	26630
5	35436	33936
End of loop	41636	

BUILD LIST

- [ ] FR-4 perfboard/protoboard, 2.54 mm grid
- [x] ~~2×8 (16-pin) 2.54 mm female header (socket for A4988)~~
- [ ] Electrolytic capacitor 100–470 µF, ≥25 V (VMOT–GND) Get from jaycar if an issue
- [ ] Ceramic capacitor 100 nF (VDD–GND) Get from jaycar if an issue
- [x] ~~2-pin screw terminal (12 V power in)~~ ~~Get from jaycar if an issue~~
- [ ] 4-pin screw terminal (motor phases)
- [x] ~~Heatsink for A4988 + thermal adhesive/paste~~
- [ ] Wire: 18–20 AWG stranded (power), 20–22 AWG stranded (motor), 22 AWG solid (on-board links)
- [x] ~~Soldering iron, solder, flux (for assembly)~~

Brown (up)

```
#define DIR_PIN 2
#define STEP_PIN 3

int stepsPerRev = 3250;       // how many steps = 1 full rotation (common value)
int stepDelay = 800;          // speed: delay between steps in microseconds

void setup() {
 pinMode(DIR_PIN, OUTPUT);
 pinMode(STEP_PIN, OUTPUT);
}

void loop() {
 // Forward 1 revolution
 digitalWrite(DIR_PIN, HIGH);
 for (int i = 0; i < stepsPerRev; i++) {
```

```
   digitalWrite(STEP_PIN, HIGH);
   delayMicroseconds(stepDelay);
   digitalWrite(STEP_PIN, LOW);
   delayMicroseconds(stepDelay);
 }
 delay(1000);

 // Backward 1 revolution
 digitalWrite(DIR_PIN, LOW);
 for (int i = 0; i < stepsPerRev; i++) {
   digitalWrite(STEP_PIN, HIGH);
   delayMicroseconds(stepDelay);
   digitalWrite(STEP_PIN, LOW);
   delayMicroseconds(stepDelay);
 }
 delay(1000);
}
```

Green up

```
#define DIR_PIN 2
#define STEP_PIN 3

int stepsPerRev = 3300;       // how many steps = 1 full rotation (common value)
int stepDelay = 1200;         // speed: delay between steps in microseconds

void setup() {
 pinMode(DIR_PIN, OUTPUT);
 pinMode(STEP_PIN, OUTPUT);
}

void loop() {
 // Forward 1 revolution
 digitalWrite(DIR_PIN, HIGH);
 for (int i = 0; i < stepsPerRev; i++) {
   digitalWrite(STEP_PIN, HIGH);
   delayMicroseconds(stepDelay);
```

```
    digitalWrite(STEP_PIN, LOW);
    delayMicroseconds(stepDelay);
  }
  delay(1000);

  // Backward 1 revolution
  digitalWrite(DIR_PIN, LOW);
  for (int i = 0; i < stepsPerRev; i++) {
    digitalWrite(STEP_PIN, HIGH);
    delayMicroseconds(stepDelay);
    digitalWrite(STEP_PIN, LOW);
    delayMicroseconds(stepDelay);
  }
  delay(1000);
}
```

Pink Down

```
#define DIR_PIN 2
#define STEP_PIN 3


int stepsPerRev = 3600;       // how many steps = 1 full rotation (common value)
int stepDelay = 5000;         // speed: delay between steps in microseconds


void setup() {
pinMode(DIR_PIN, OUTPUT);
pinMode(STEP_PIN, OUTPUT);
}


void loop() {
// Forward 1 revolution
digitalWrite(DIR_PIN, HIGH);
```

```
for (int i = 0; i < stepsPerRev; i++) {
  digitalWrite(STEP_PIN, HIGH);
  delayMicroseconds(stepDelay);
  digitalWrite(STEP_PIN, LOW);
  delayMicroseconds(stepDelay);
}
delay(1000);


// Backward 1 revolution
digitalWrite(DIR_PIN, LOW);
for (int i = 0; i < stepsPerRev; i++) {
  digitalWrite(STEP_PIN, HIGH);
  delayMicroseconds(stepDelay);
  digitalWrite(STEP_PIN, LOW);
  delayMicroseconds(stepDelay);
}
delay(1000);
}
```

Green floor

```
#define DIR_PIN 2
#define STEP_PIN 3


int stepsPerRev = 3600;       // how many steps = 1 full rotation (common value)
int stepDelay = 800;         // speed: delay between steps in microseconds


void setup() {
pinMode(DIR_PIN, OUTPUT);
pinMode(STEP_PIN, OUTPUT);
}
```

```
void loop() {
// Forward 1 revolution
digitalWrite(DIR_PIN, HIGH);
for (int i = 0; i < stepsPerRev; i++) {
  digitalWrite(STEP_PIN, HIGH);
  delayMicroseconds(stepDelay);
  digitalWrite(STEP_PIN, LOW);
  delayMicroseconds(stepDelay);
}
delay(1000);


// Backward 1 revolution
digitalWrite(DIR_PIN, LOW);
for (int i = 0; i < stepsPerRev; i++) {
  digitalWrite(STEP_PIN, HIGH);
  delayMicroseconds(stepDelay);
  digitalWrite(STEP_PIN, LOW);
  delayMicroseconds(stepDelay);
}
delay(1000);
}
```

Light blue up

```
#define DIR_PIN 2
#define STEP_PIN 3
```

```
int stepsPerRev = 3000;       // how many steps = 1 full rotation (common value)
int stepDelay = 3000;         // speed: delay between steps in microseconds



void setup() {
pinMode(DIR_PIN, OUTPUT);
pinMode(STEP_PIN, OUTPUT);
}



void loop() {
// Forward 1 revolution
digitalWrite(DIR_PIN, HIGH);
for (int i = 0; i < stepsPerRev; i++) {
 digitalWrite(STEP_PIN, HIGH);
 delayMicroseconds(stepDelay);
 digitalWrite(STEP_PIN, LOW);
 delayMicroseconds(stepDelay);
}
delay(1000);



// Backward 1 revolution
digitalWrite(DIR_PIN, LOW);
for (int i = 0; i < stepsPerRev; i++) {
 digitalWrite(STEP_PIN, HIGH);
 delayMicroseconds(stepDelay);
 digitalWrite(STEP_PIN, LOW);
 delayMicroseconds(stepDelay);
}
delay(1000);
}
```