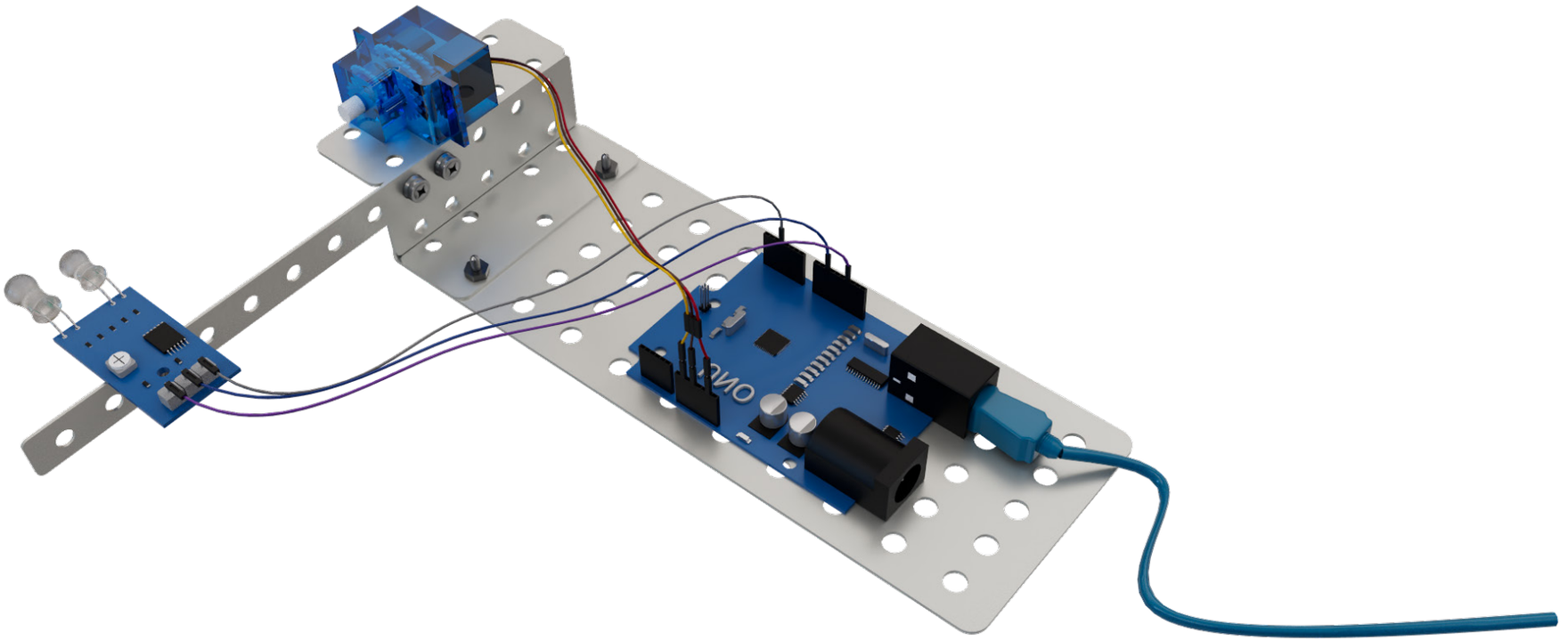
MECHANZ™

Automatic Gate

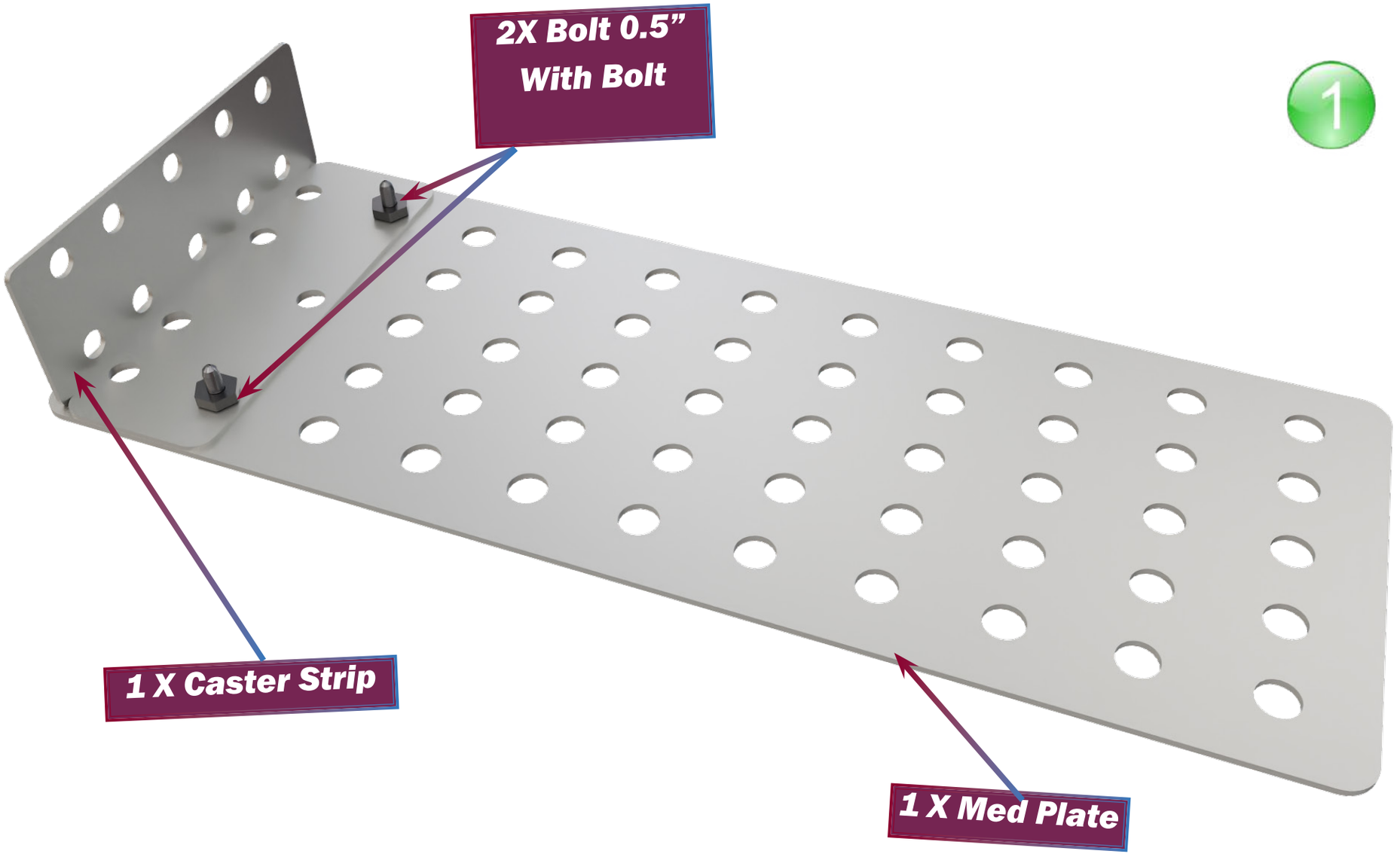


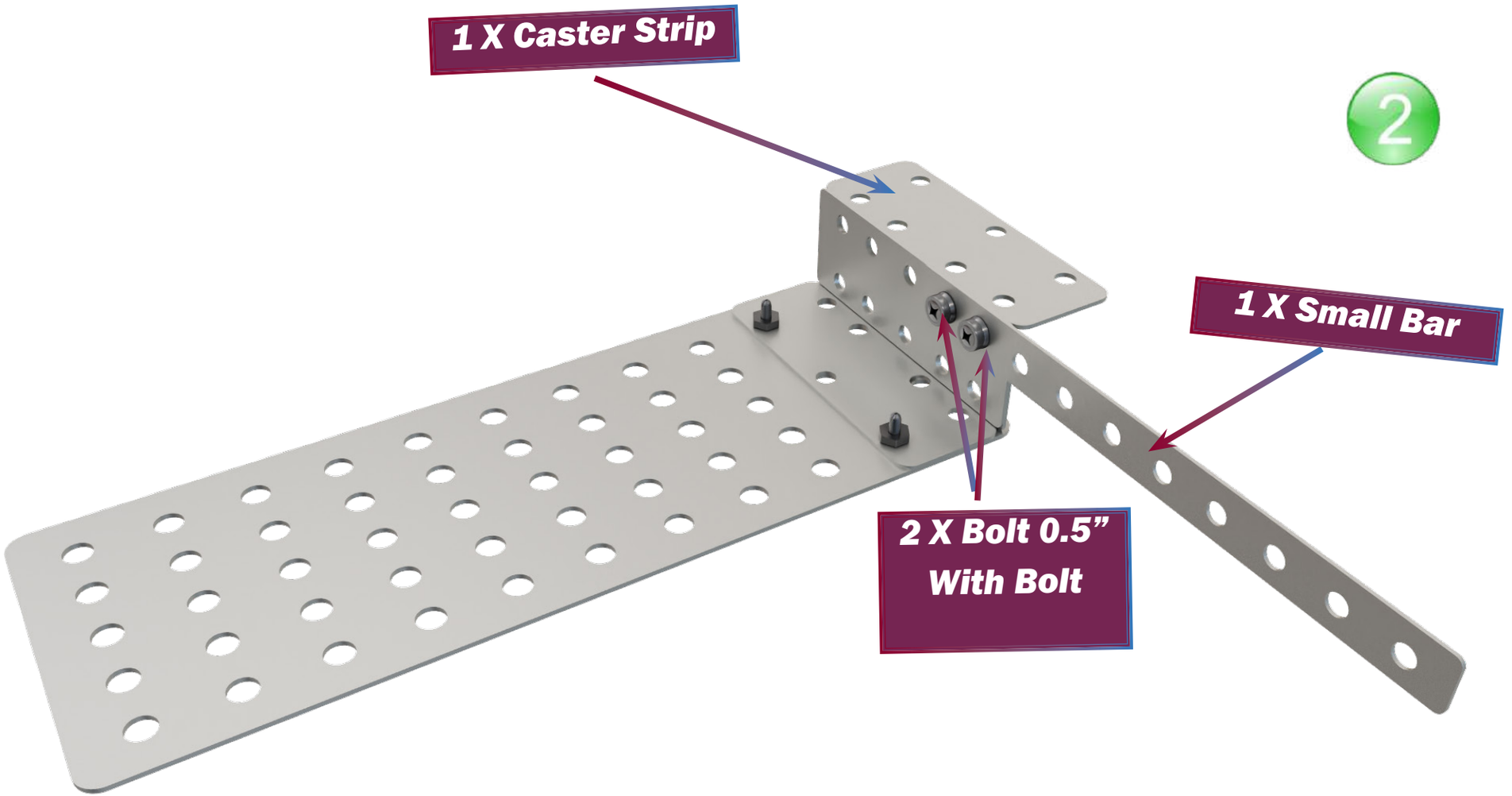
Product of

ABLEDUCATION™

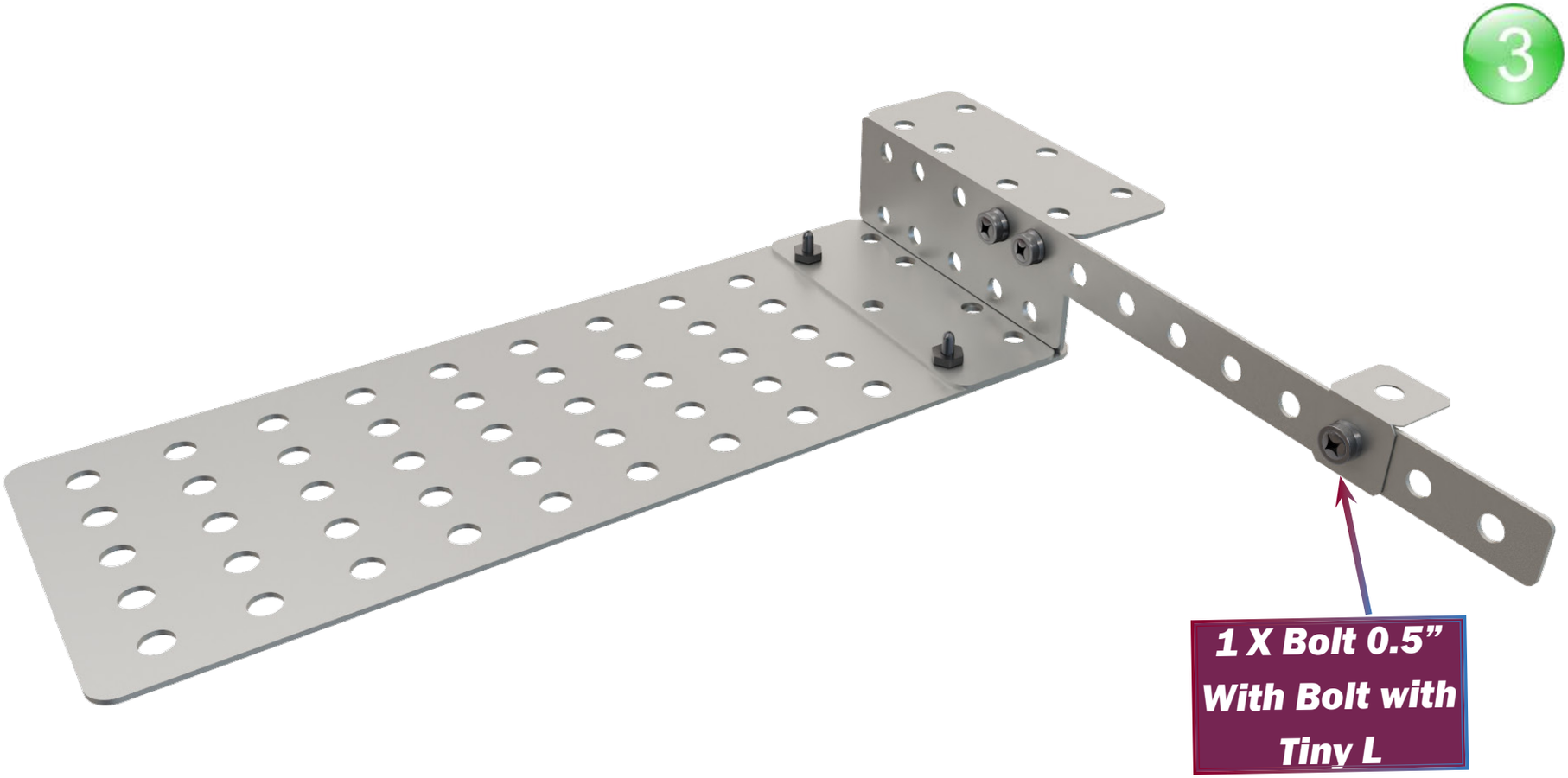
THINK • EXPLORE • CREATE • INNOVATE

MECHANZ™

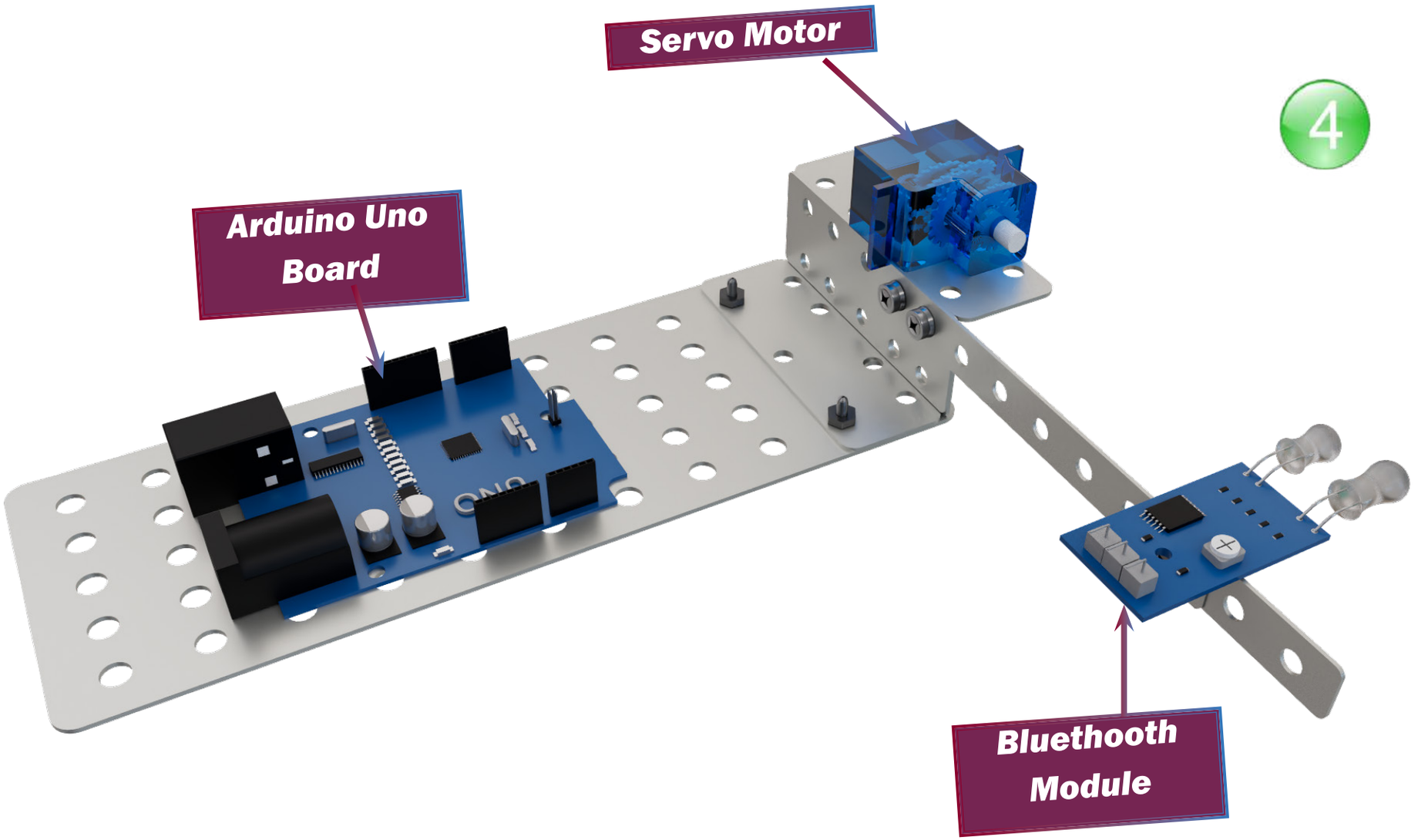




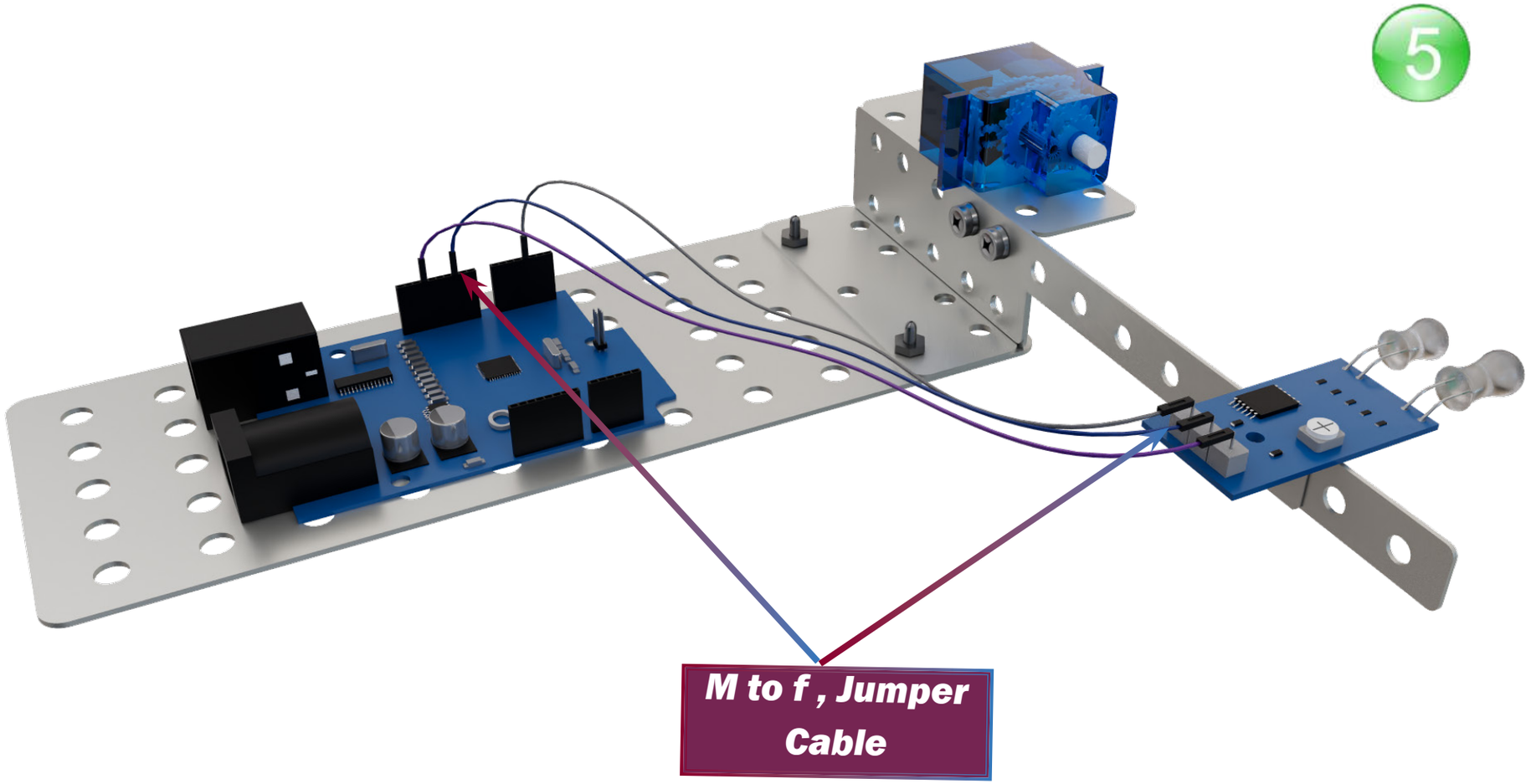
MECHANZ™



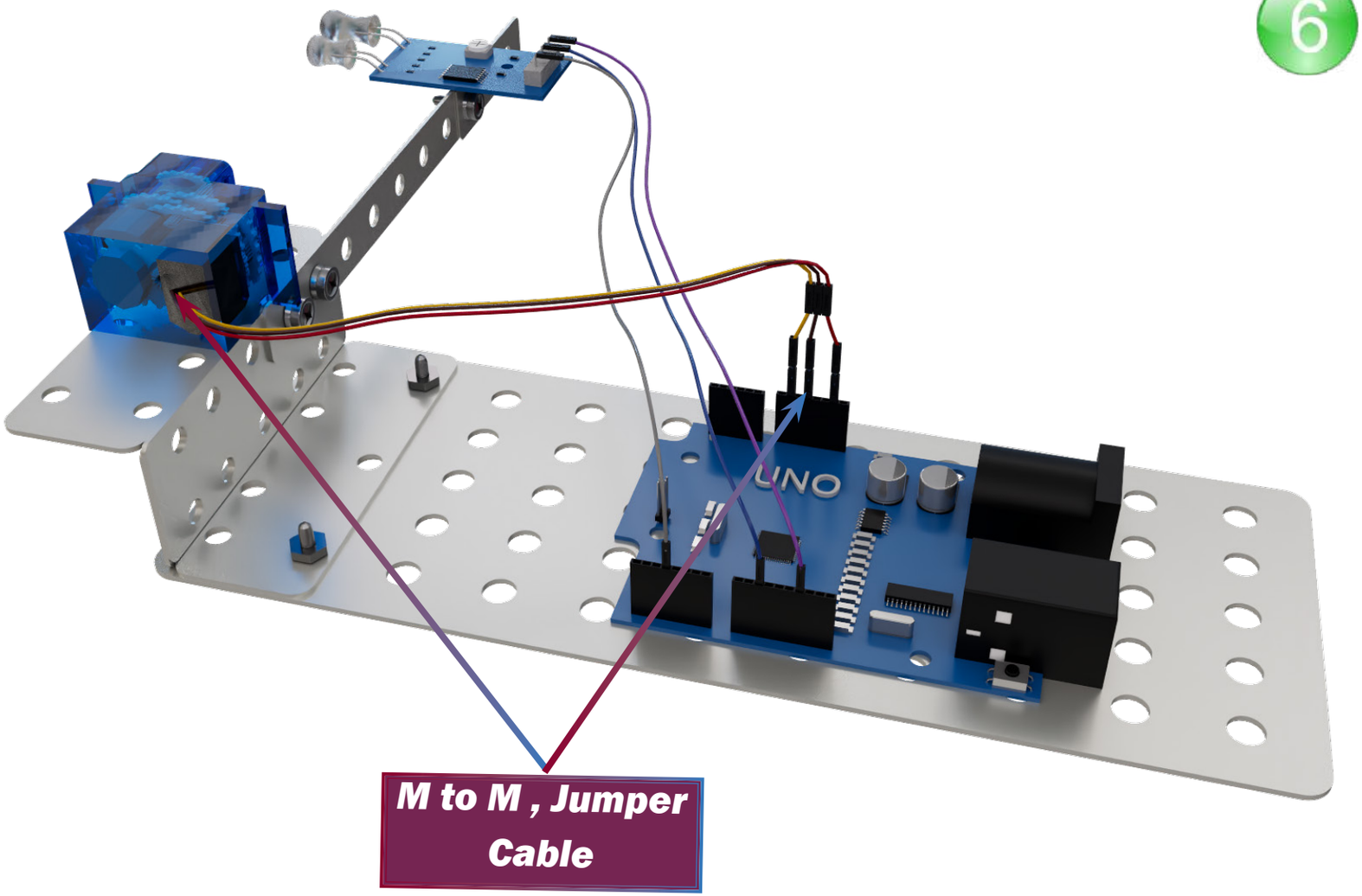
MECHANZ™



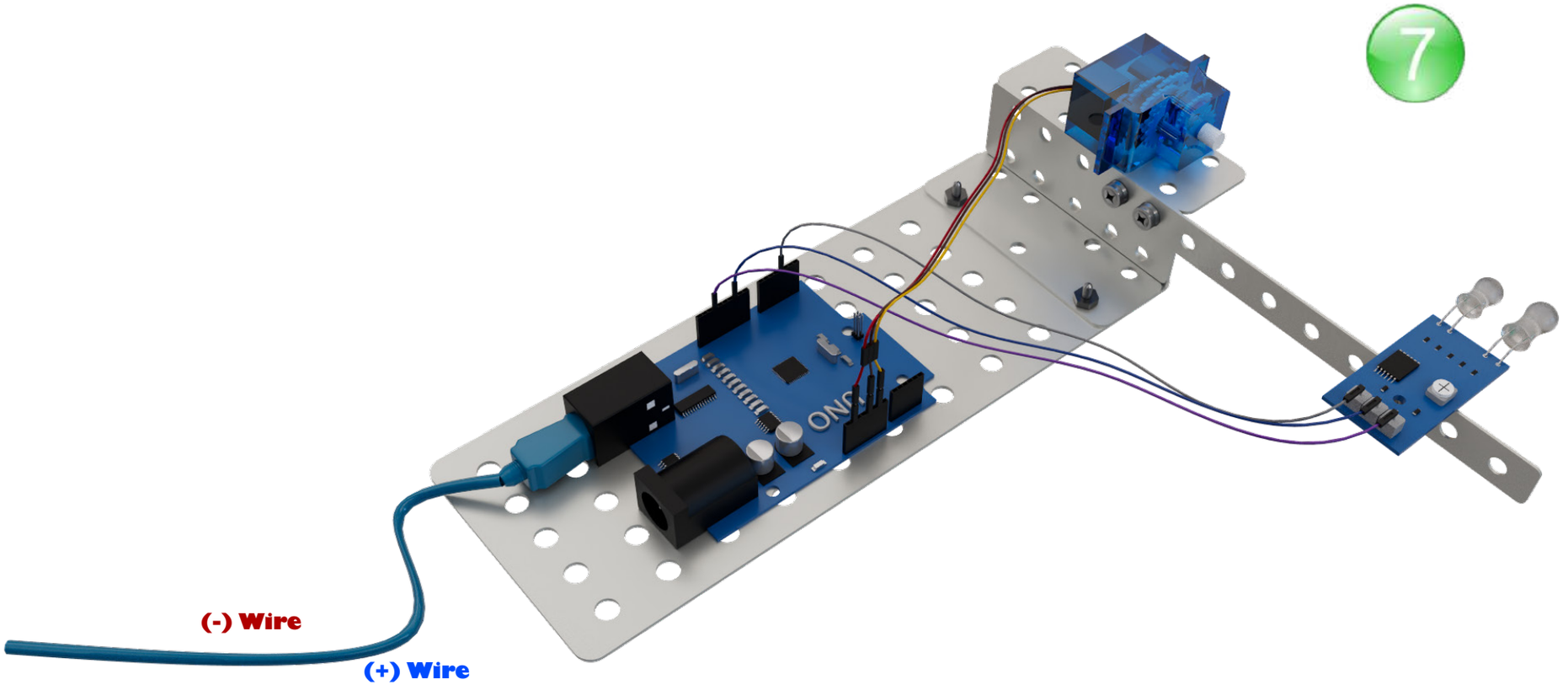
MECHANZ™



6



MECHANZ™



Power Supply from 12 V battery

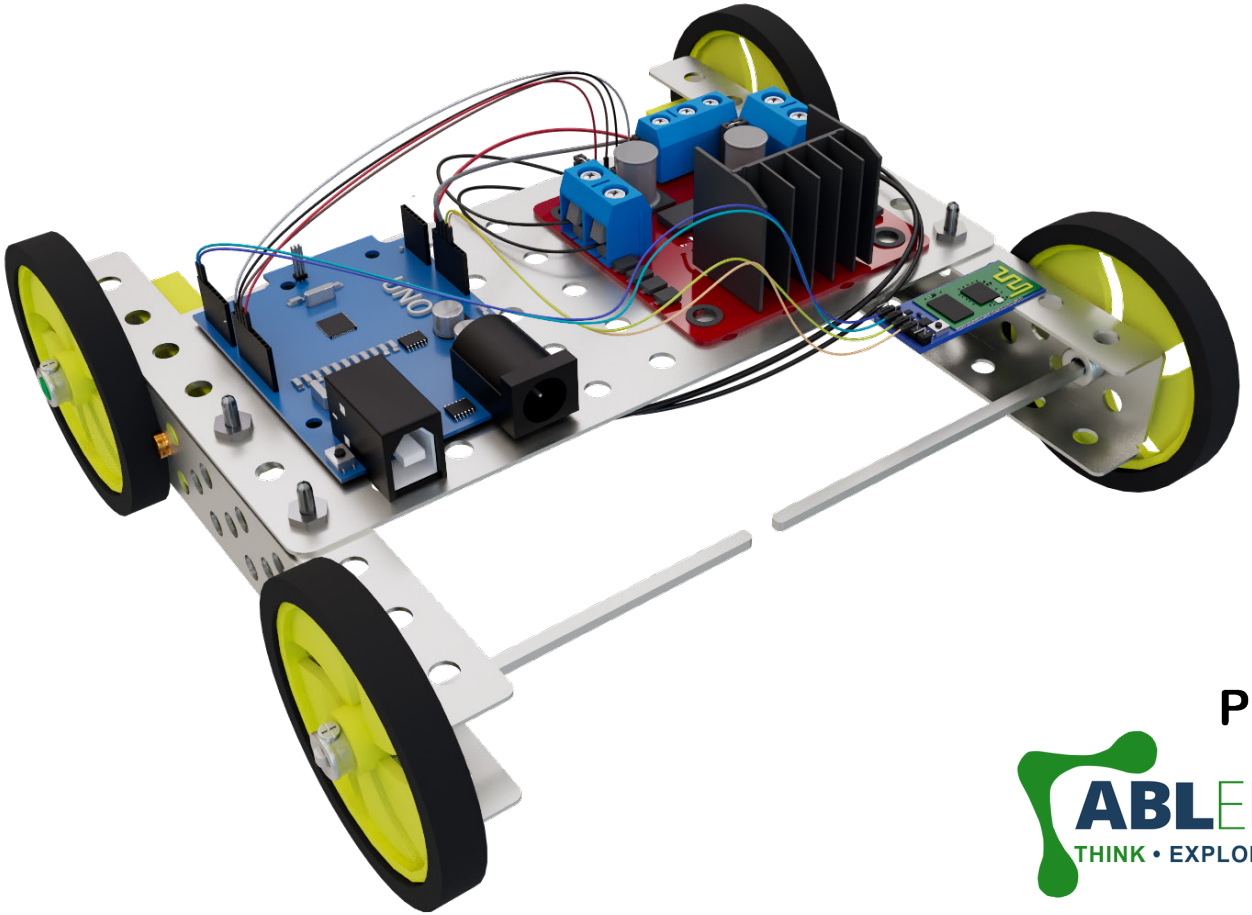
Coding For Automatic Gate

```
#include <Servo.h> // servo library
 Servo s1;
 int val = 0 ;
 int va2 = 0 ;
 void setup()
 {
   Serial.begin(9600); // sensor buart rate
   pinMode(2,INPUT);  // IR sensor 1
   pinMode(3,INPUT);  // IR sensor 2
   s1.attach(9);    // Servo Connect 9 pin
   s1.write(0);
 }
 void loop()
 {
  val = digitalRead(2); // IR sensor 1 output pin
connected
  va2 = digitalRead(3); // IR sensor 2 output pin
connected
  Serial.println(val); // see the value in serial
mpnitor in Arduino IDE
  Serial.println(va2); // see the value in serial
mpnitor in Arduino IDE
  delay(10);      // Time Delay
```

```
if(val == 1  )
 {
 s1.write(0); // SERVO 0 DEGREE
 }
if(val == 0 )
 {
 s1.write(90);
 delay(5000);
 s1.write(0);// SERVO 90 DEGREE
 }
 if(val==1 )
 { s1.write(0);
 }
}
```

MECHANZ™

Bluetooth Control Bot

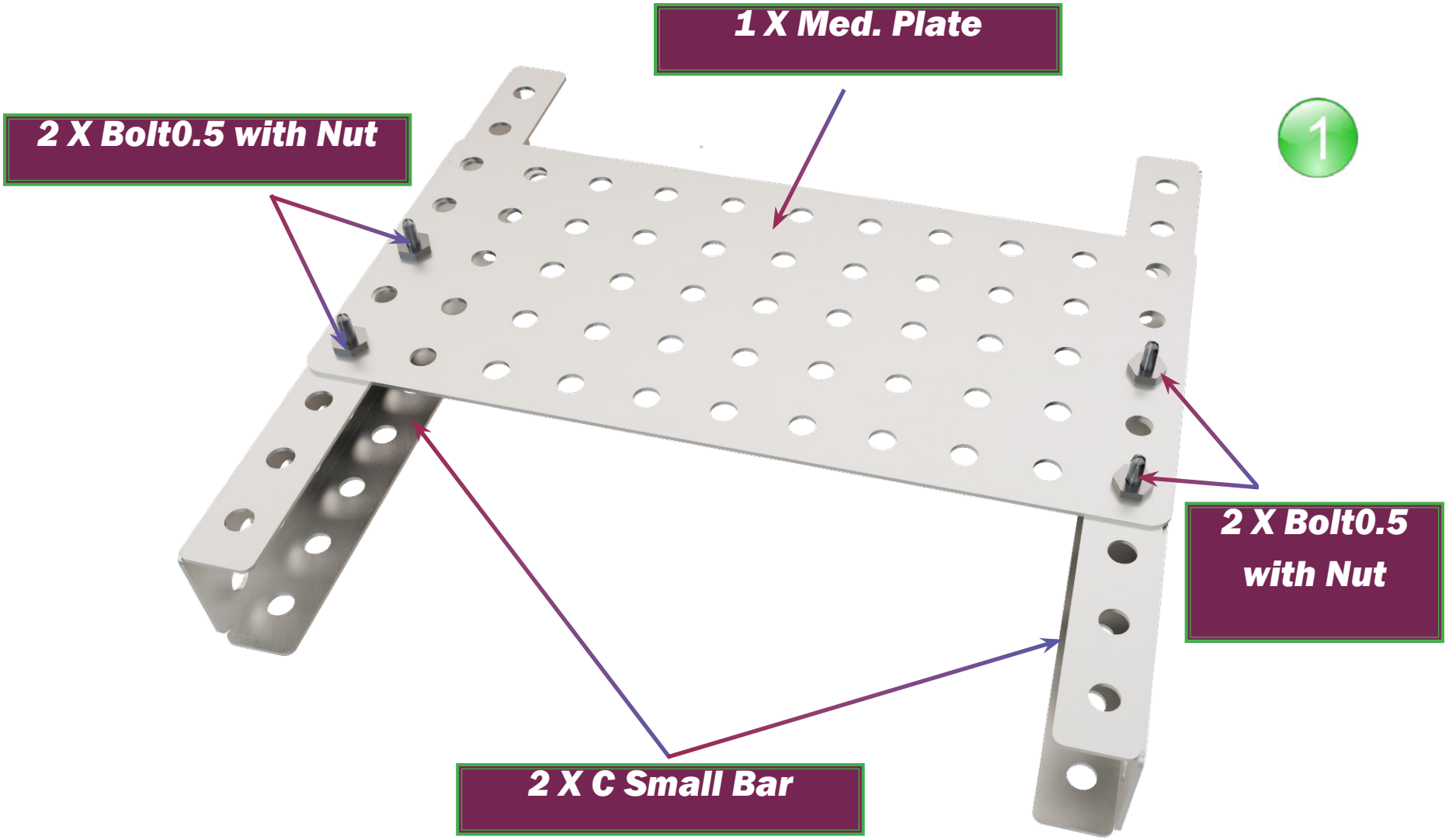


Product of

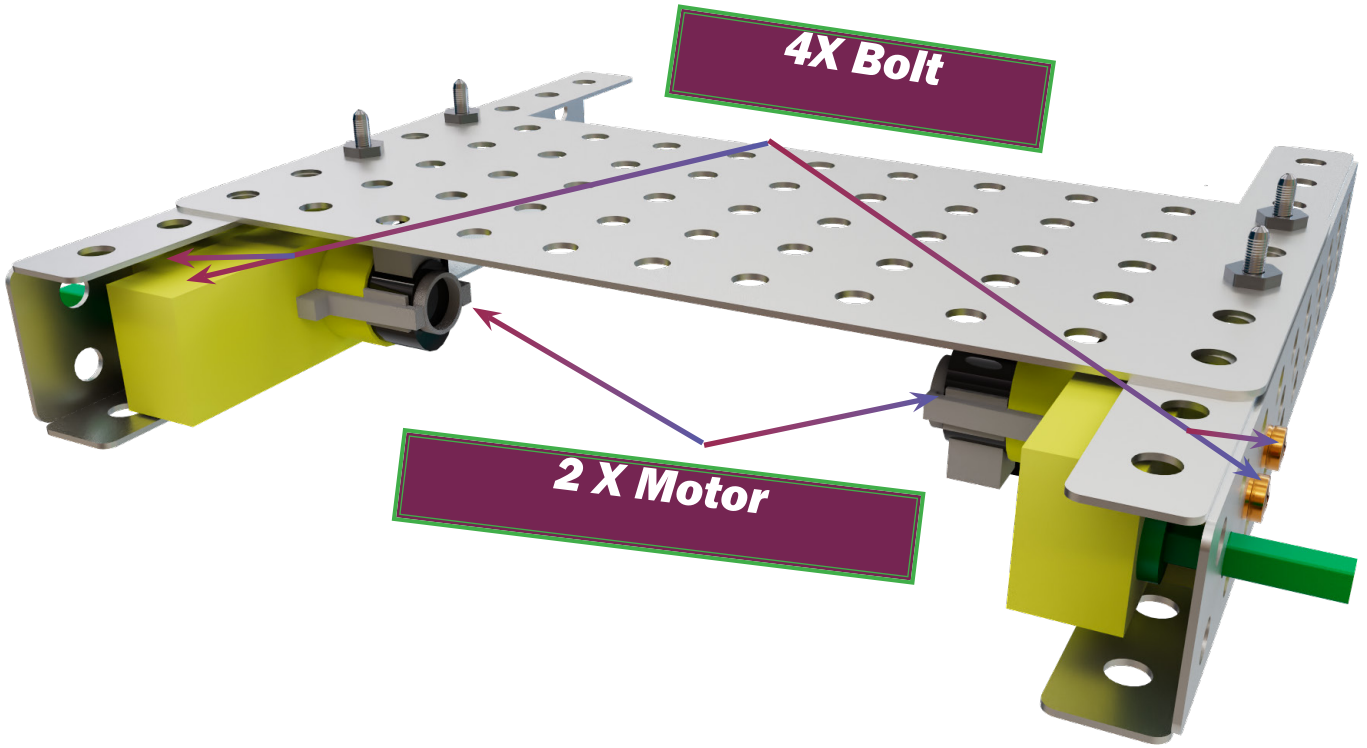
ABLEDUCATION™

THINK • EXPLORE • CREATE • INNOVATE

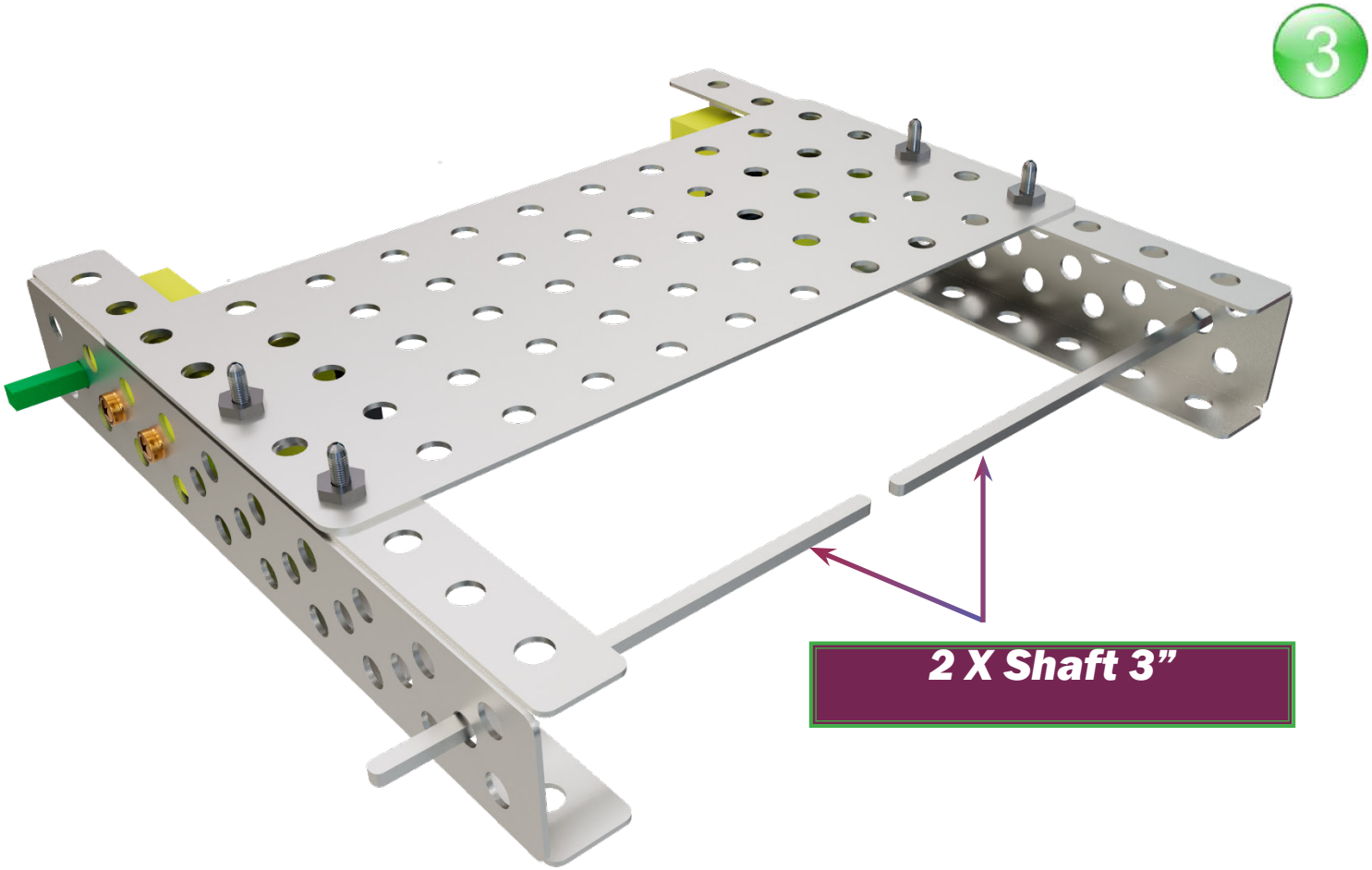
MECHANZ™

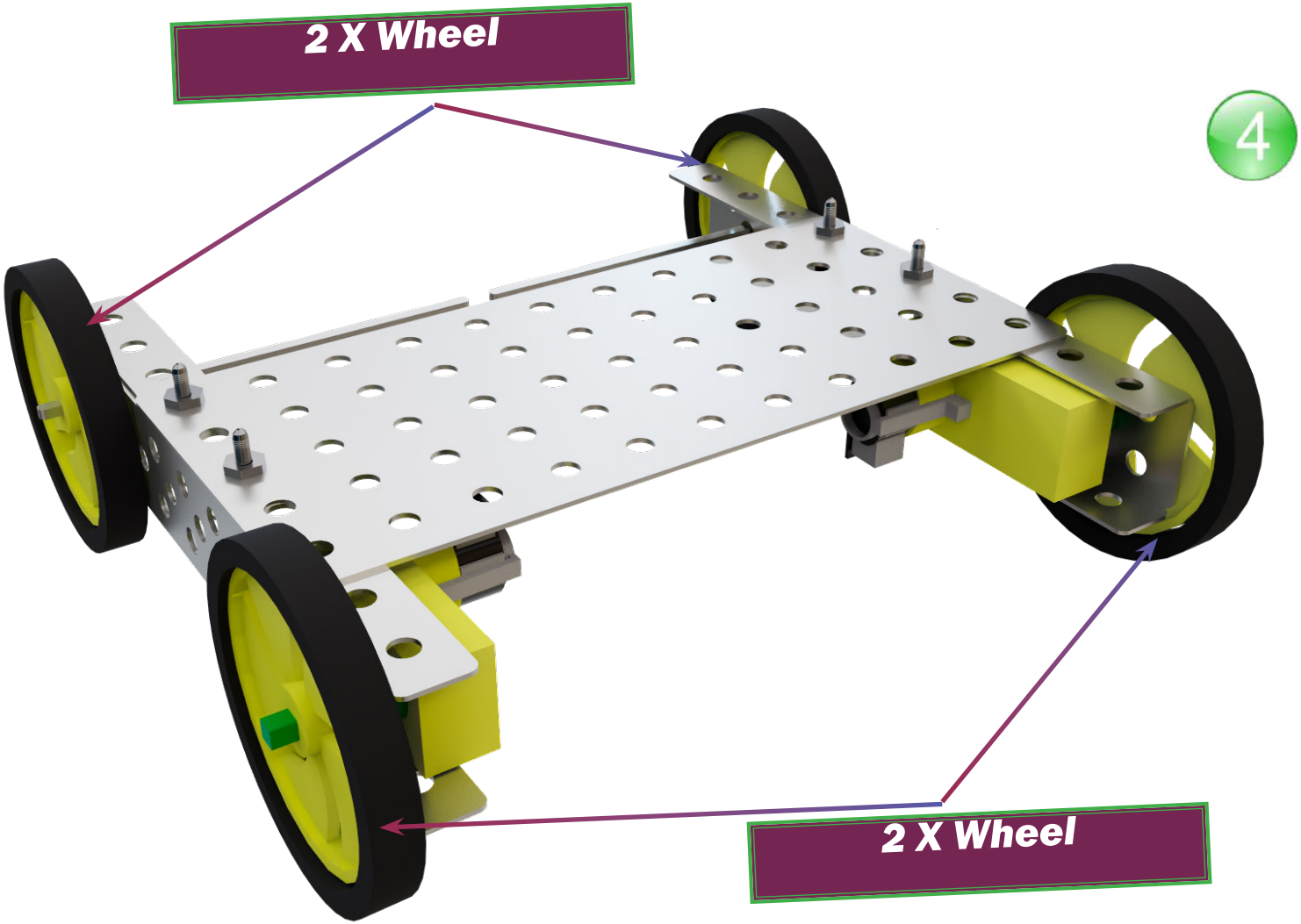


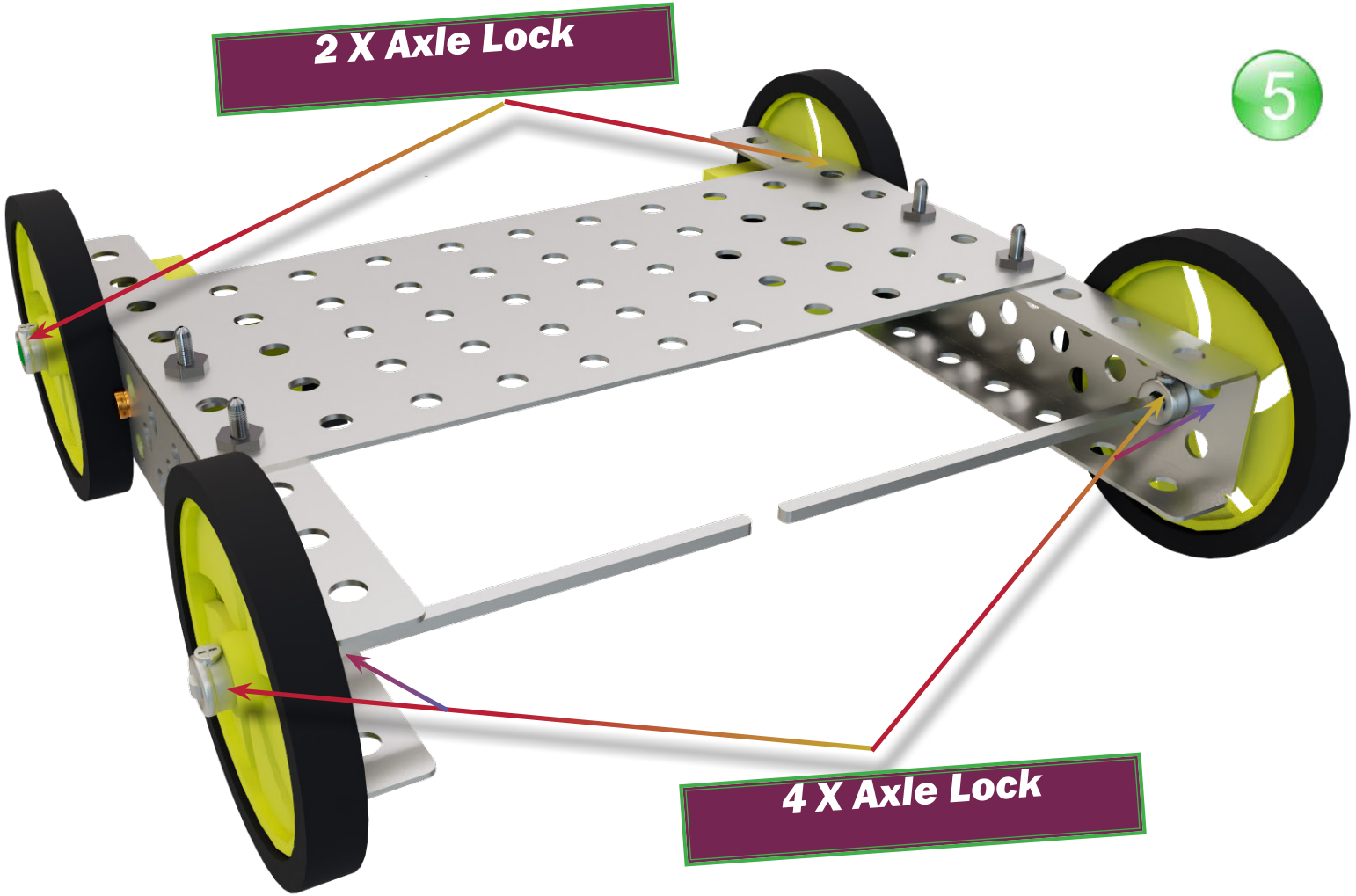
2



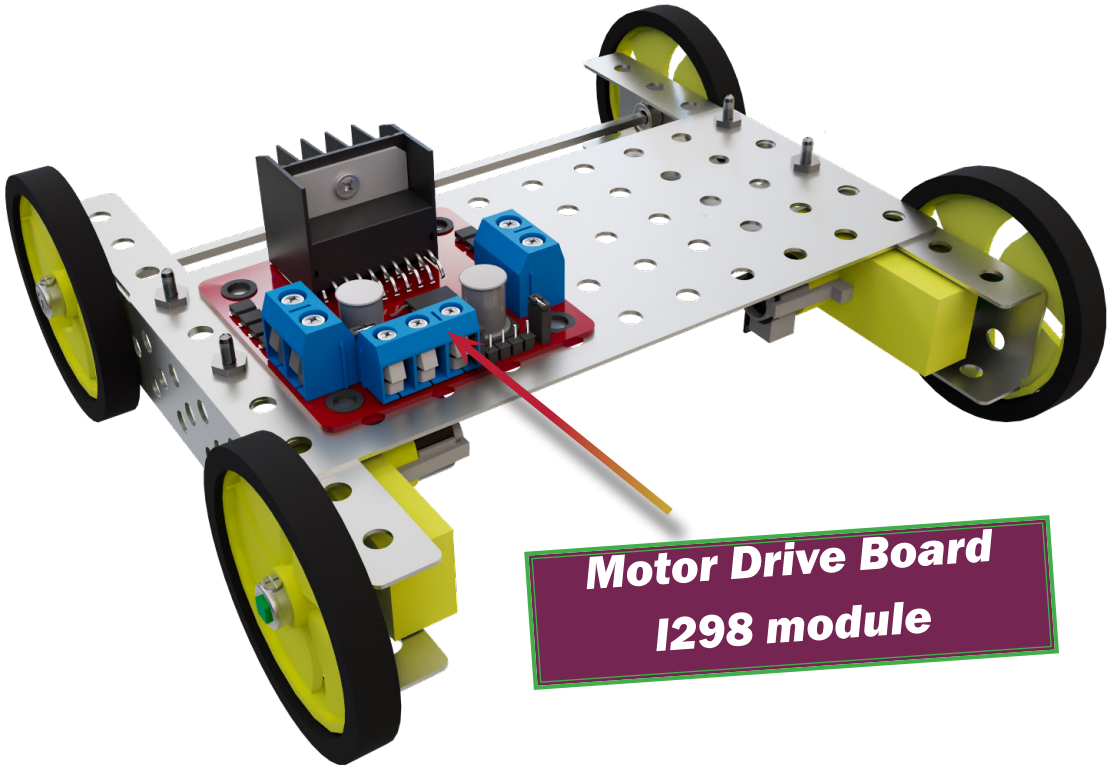
MECHANZ™



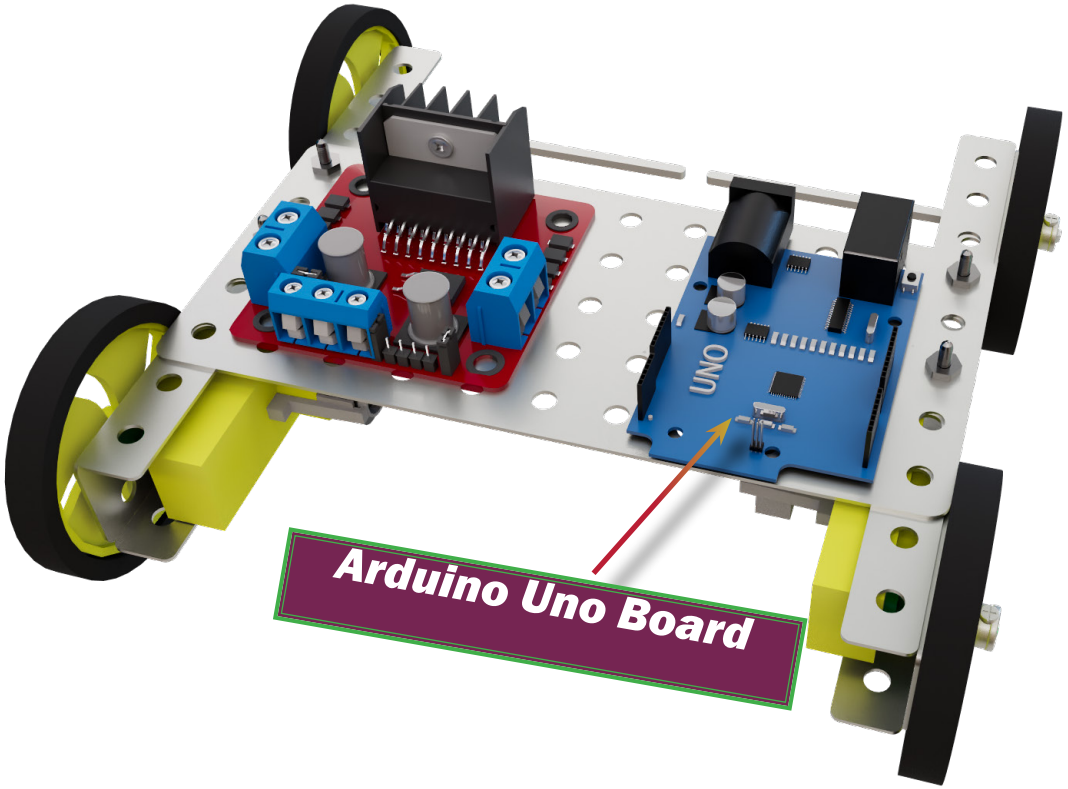




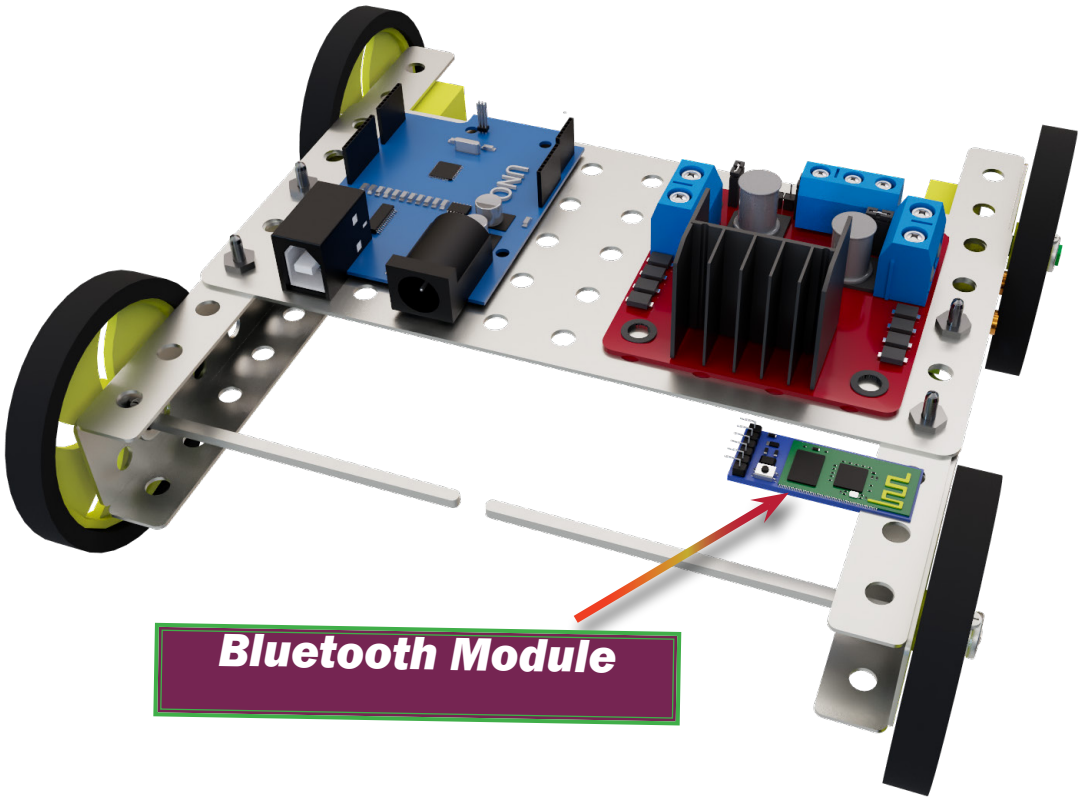
6



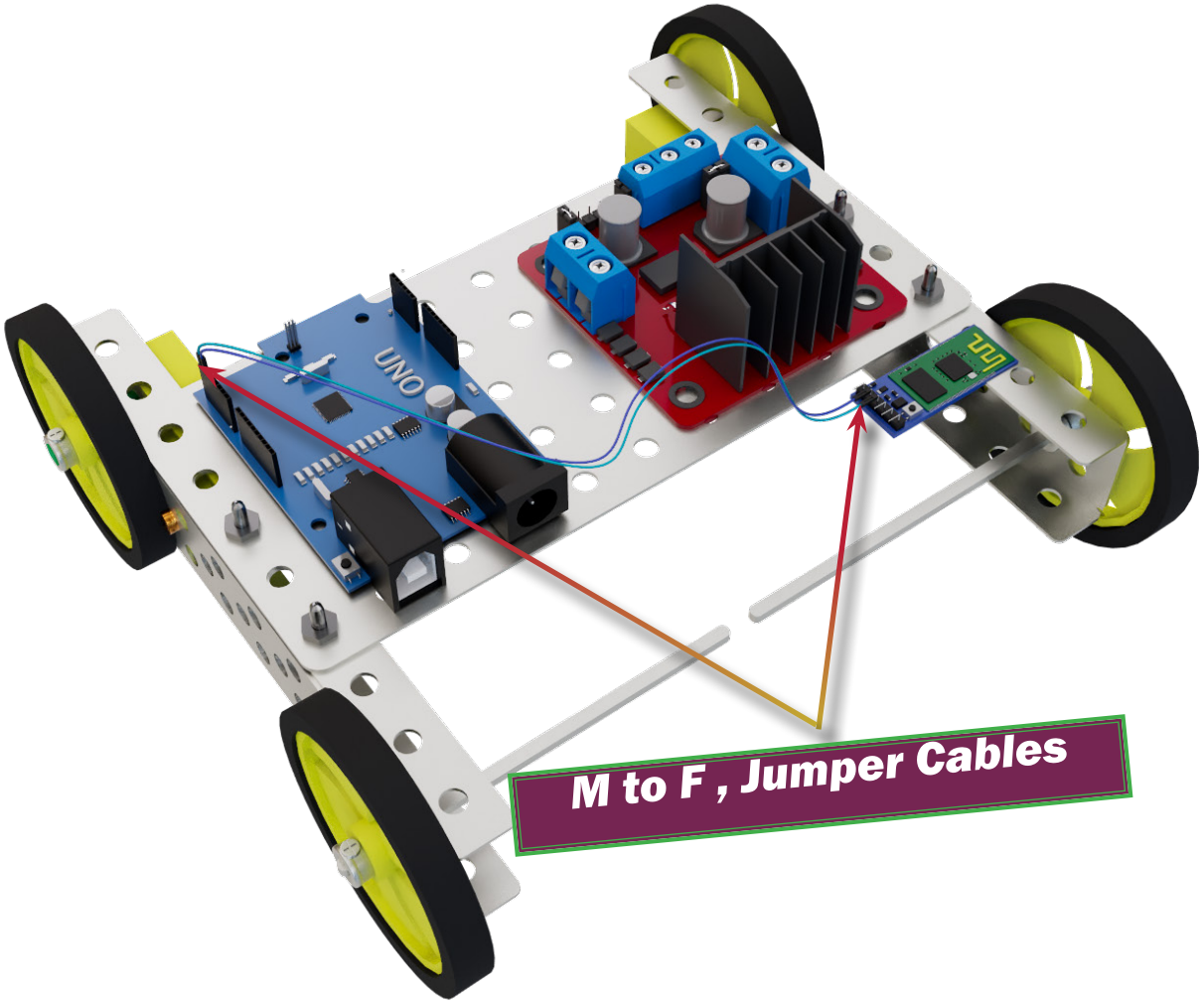
7



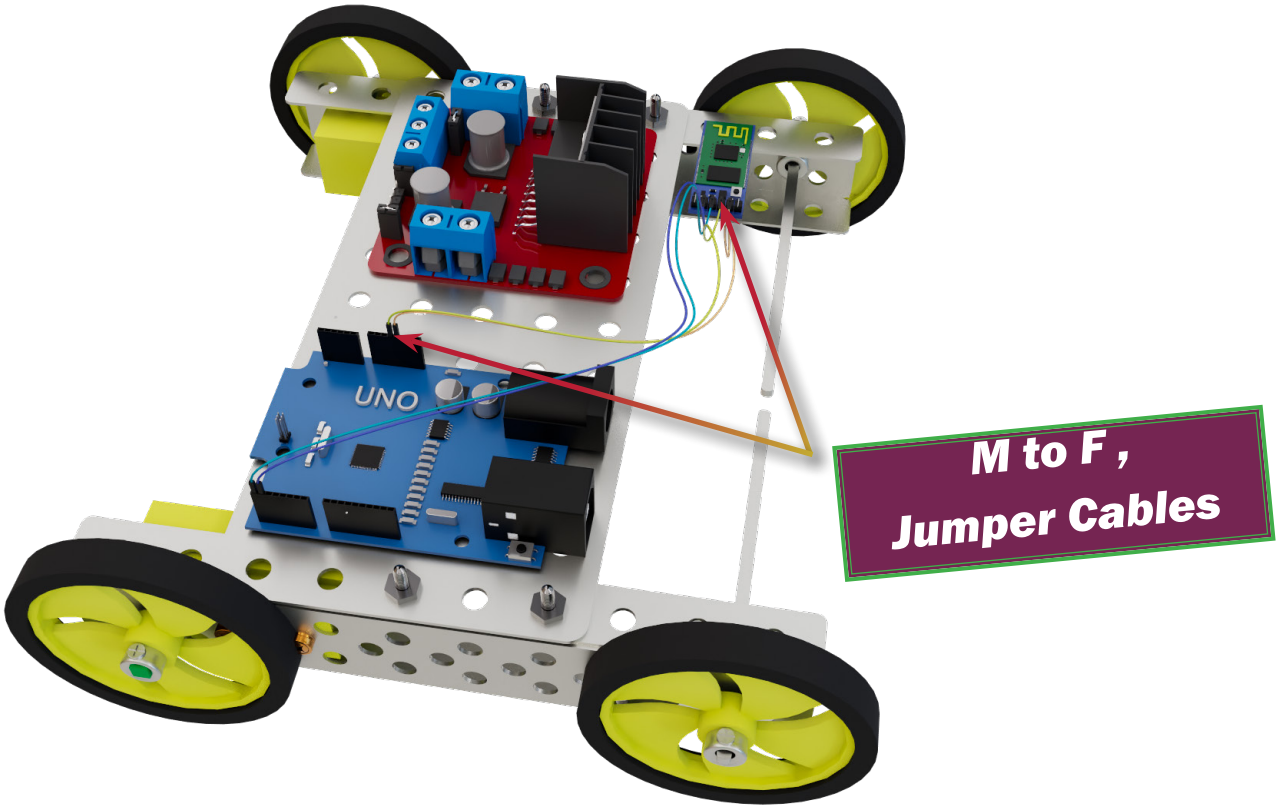
8

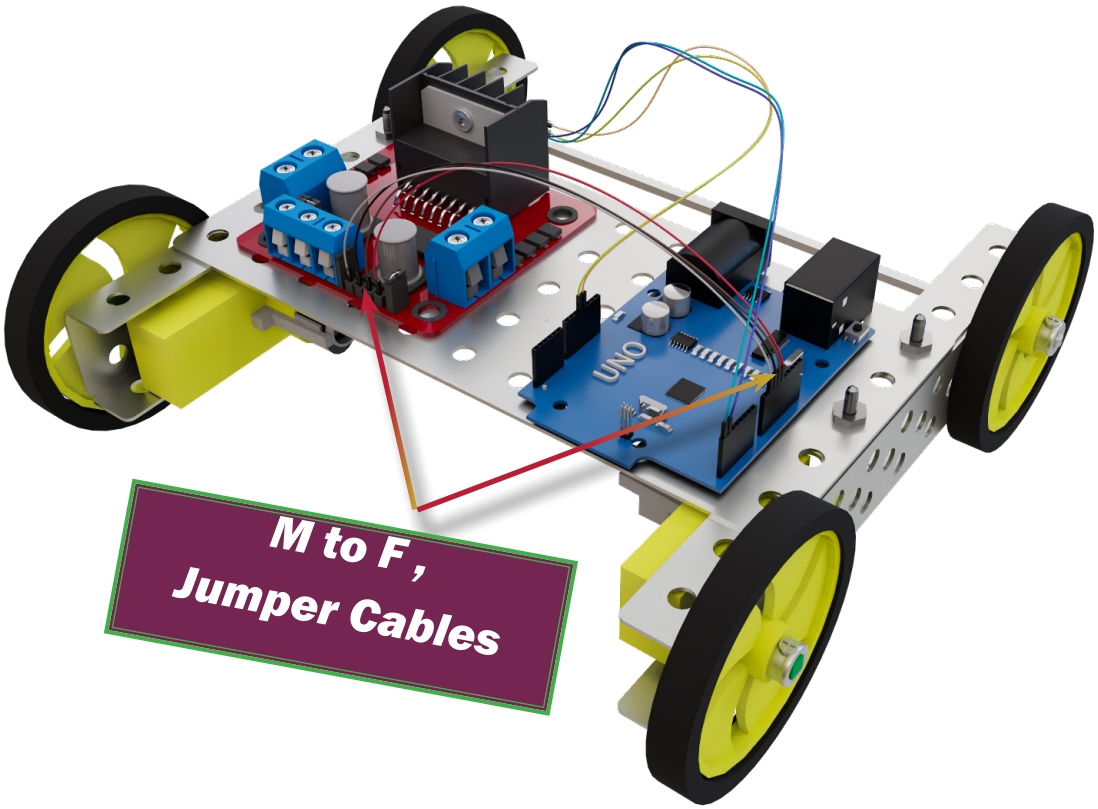


MECHANZ™

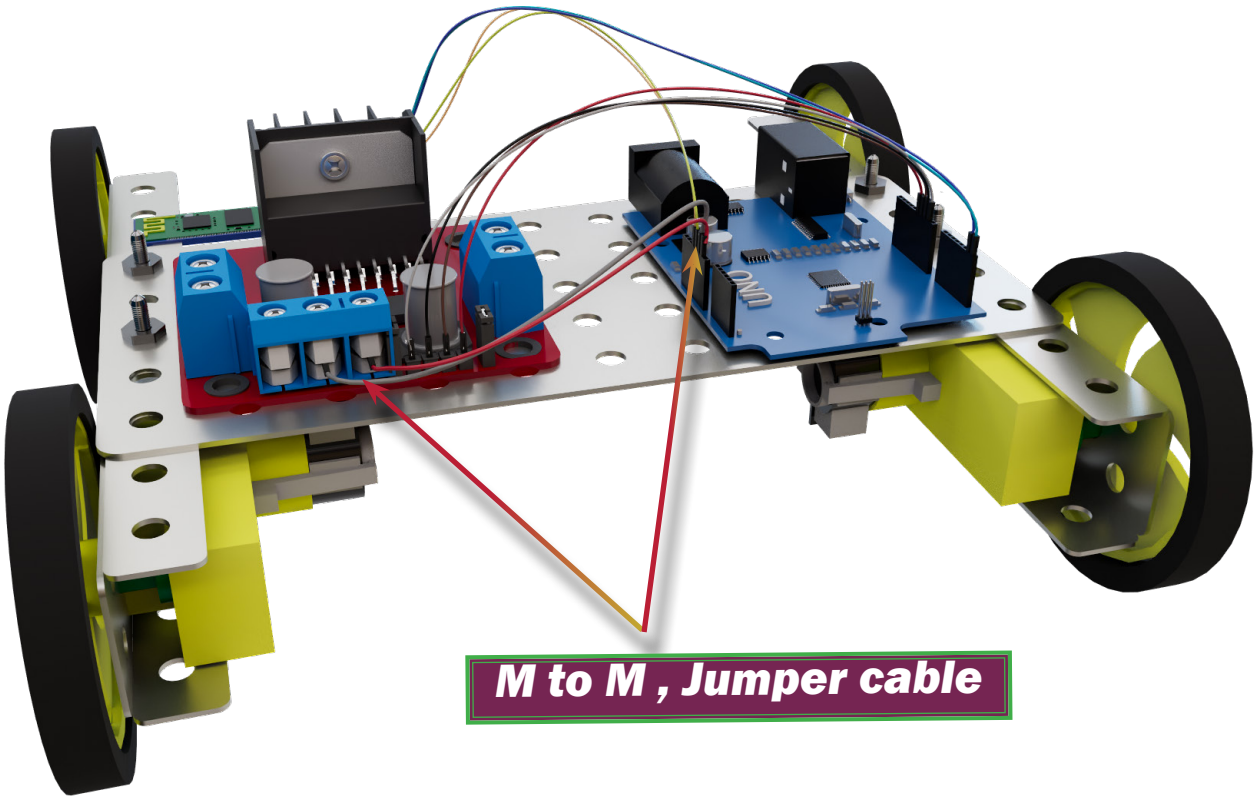


10

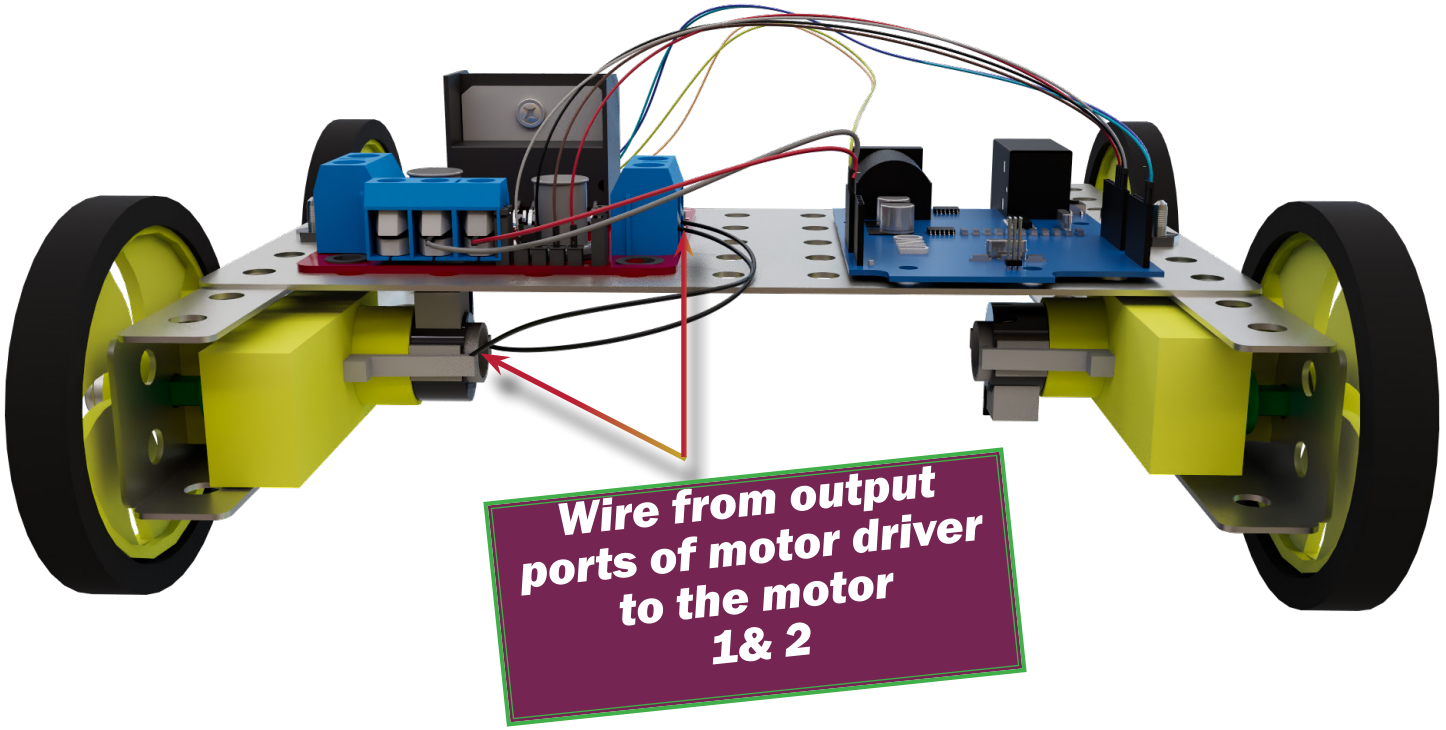


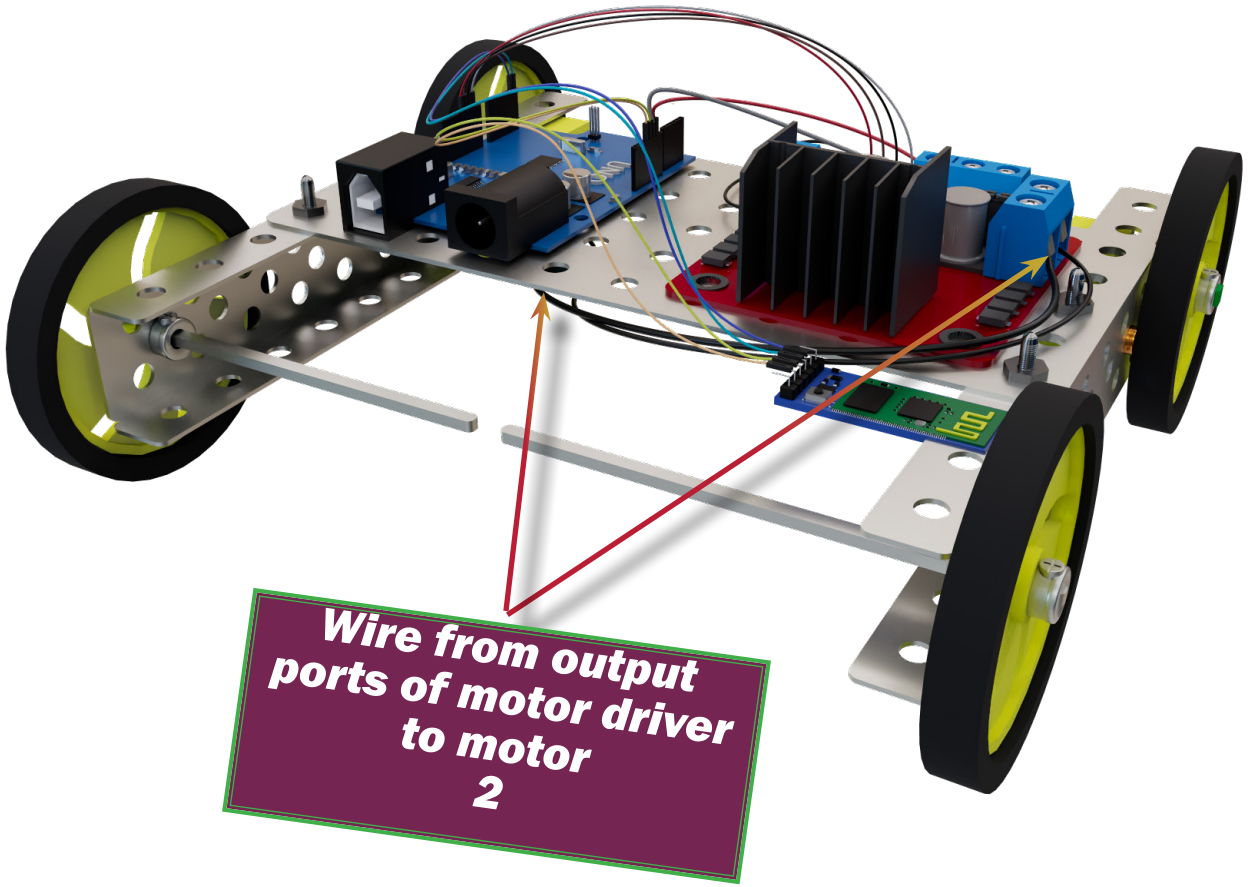


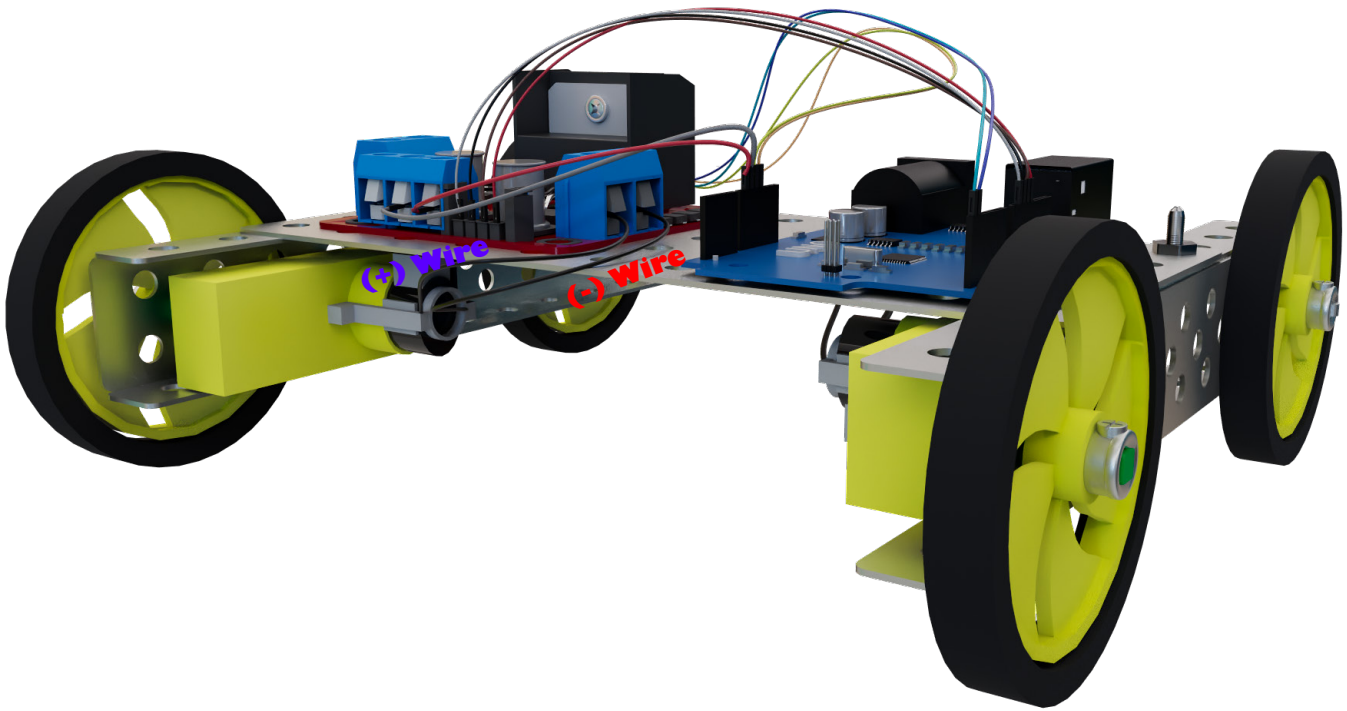
12



13



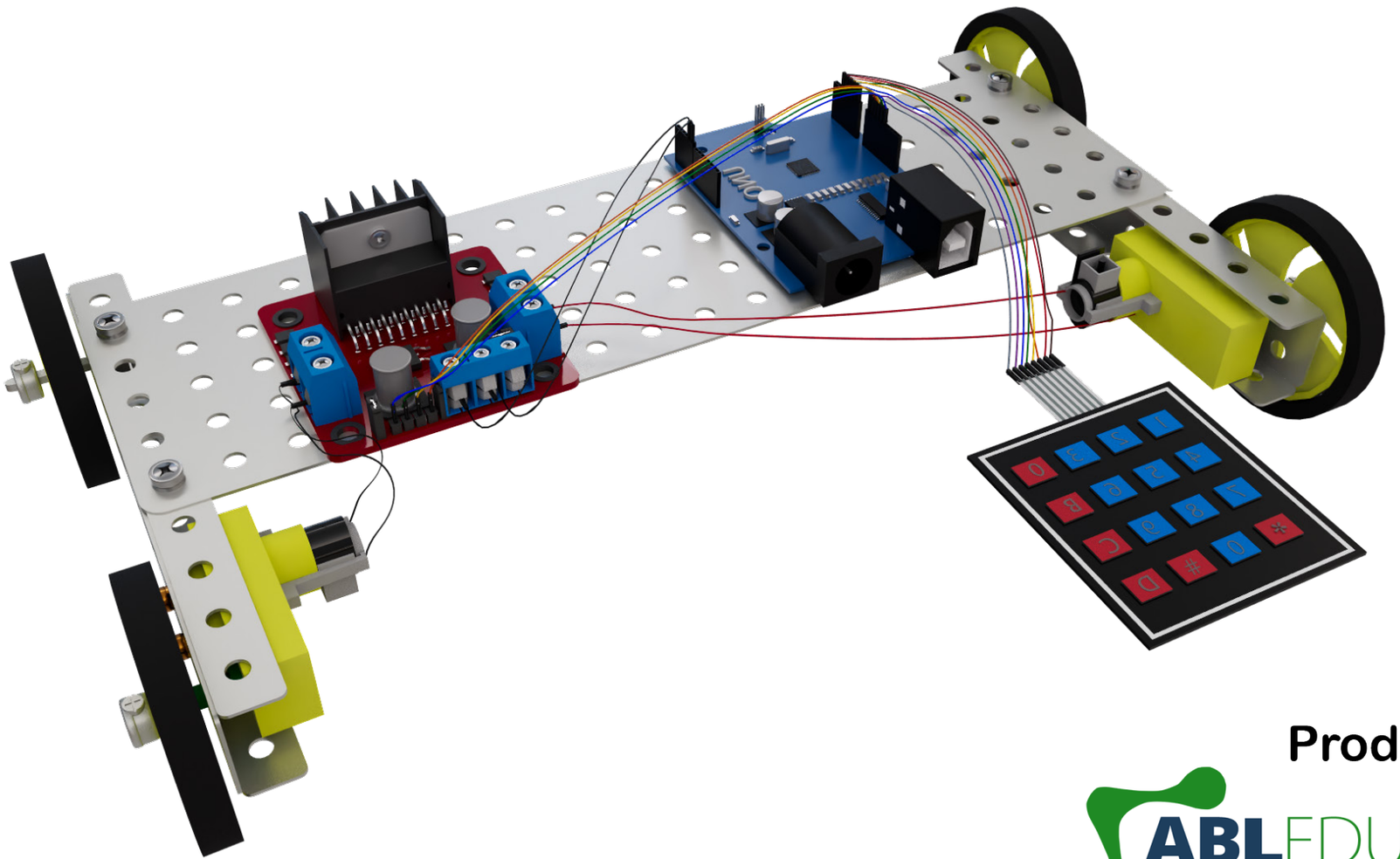




*** Connection of 9V battery to power pins of Motor driver**

MECHANZ™

Keypad Control Bot

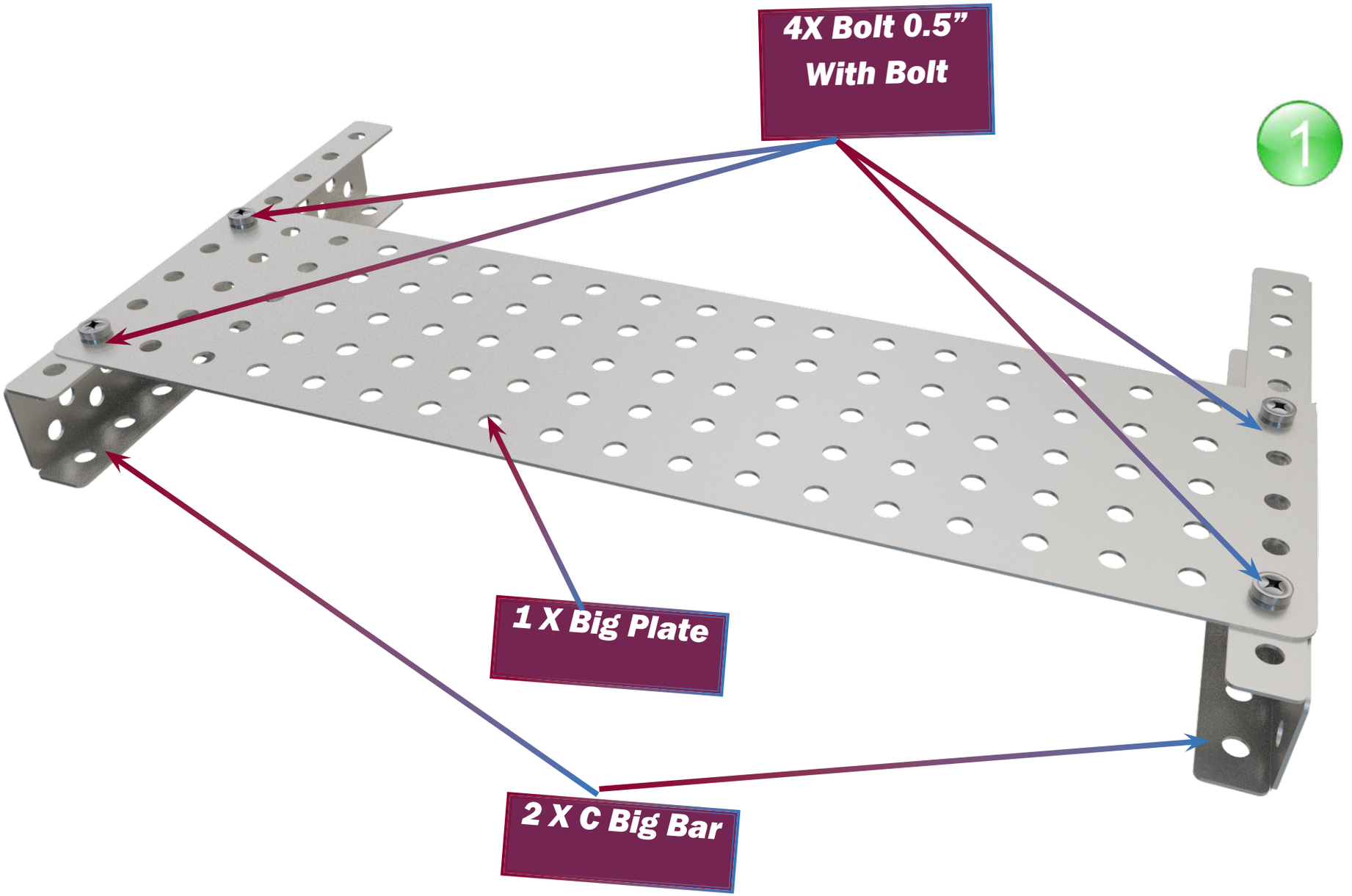


Product of

ABLEDUCATION™

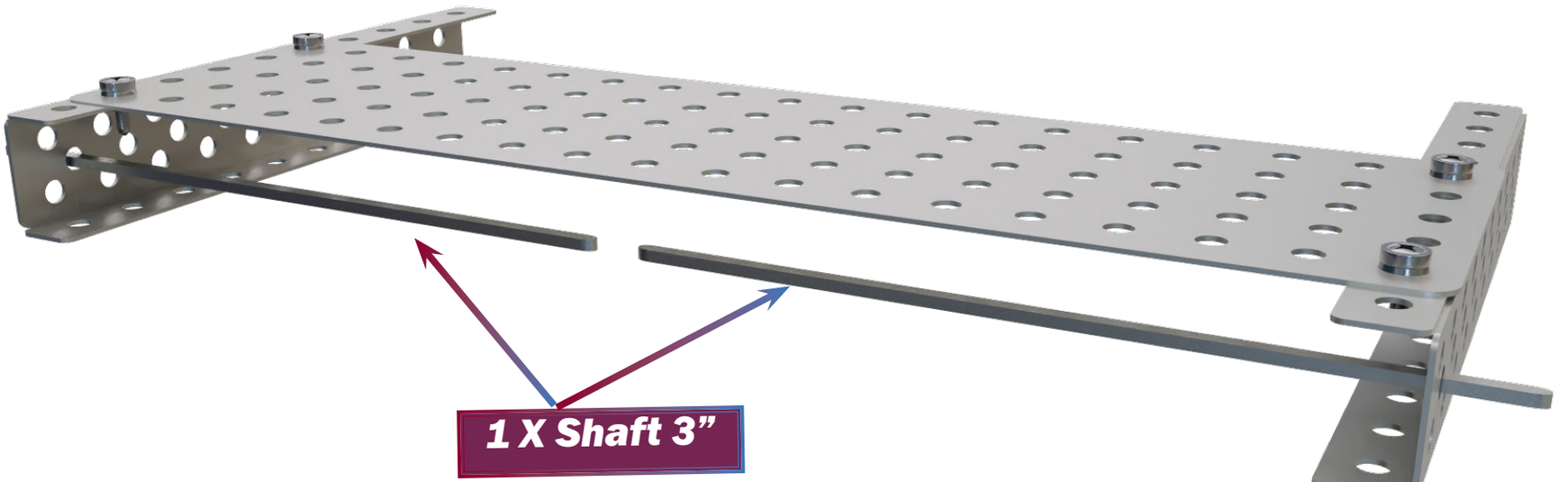
THINK • EXPLORE • CREATE • INNOVATE

MECHANZ™

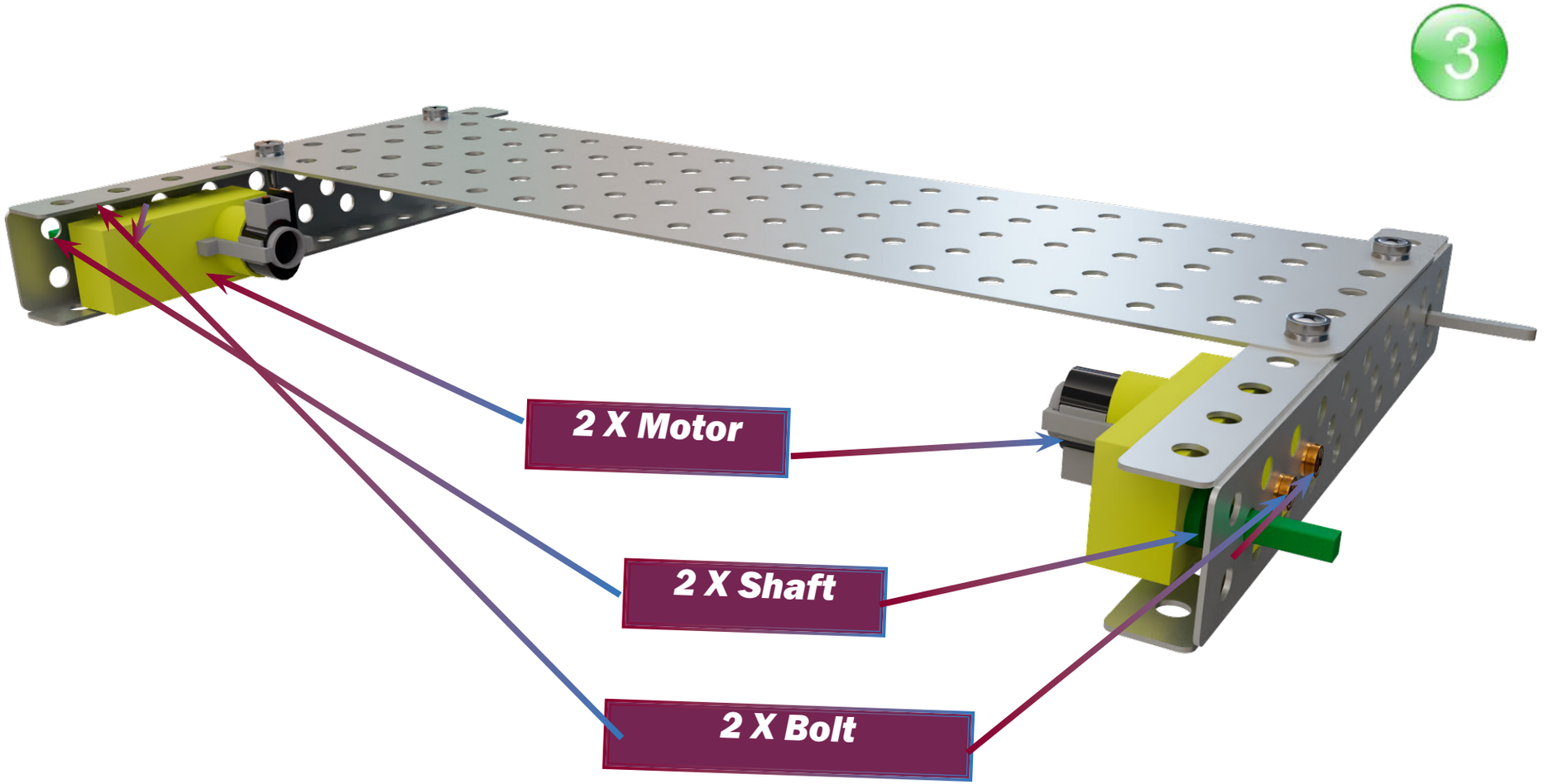


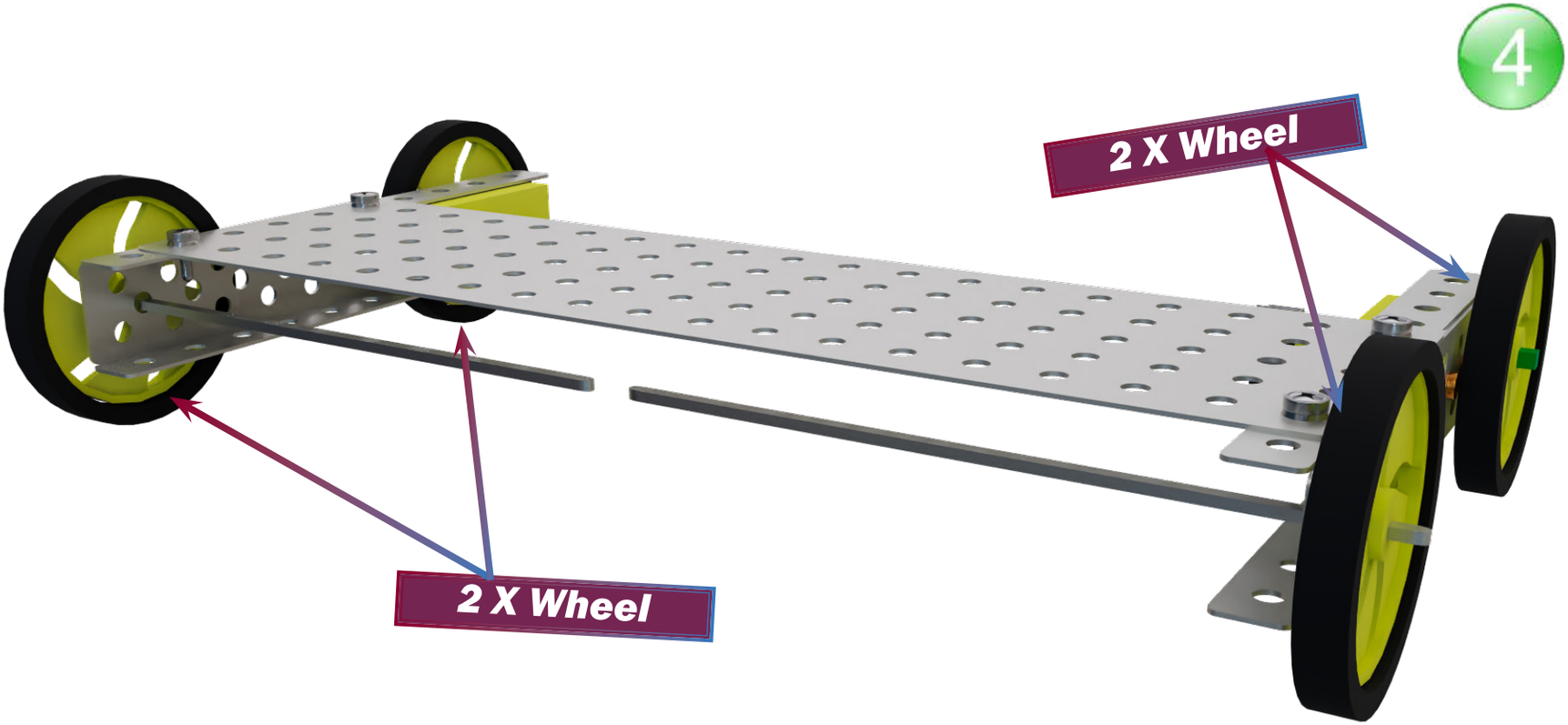
MECHANZ™

2

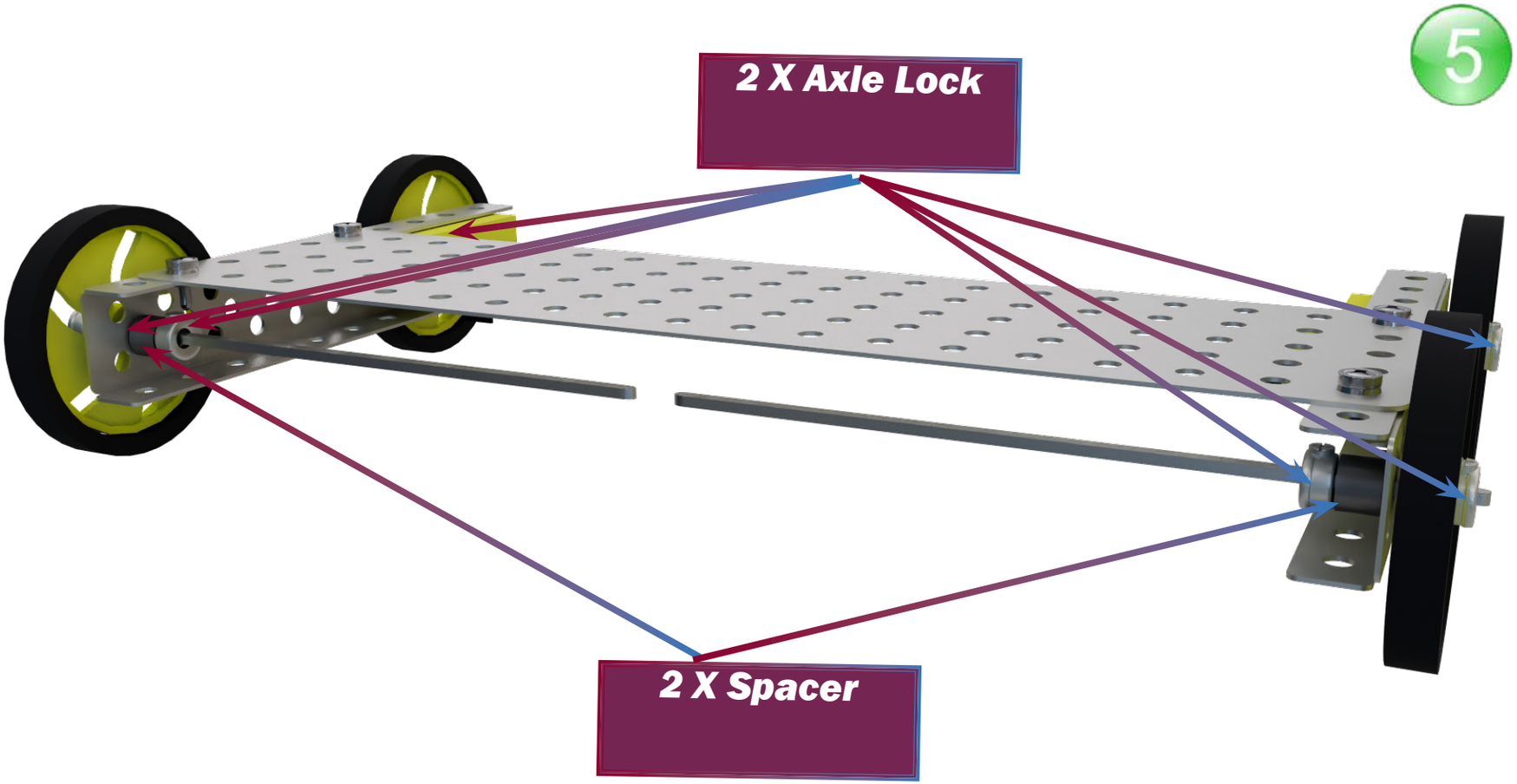


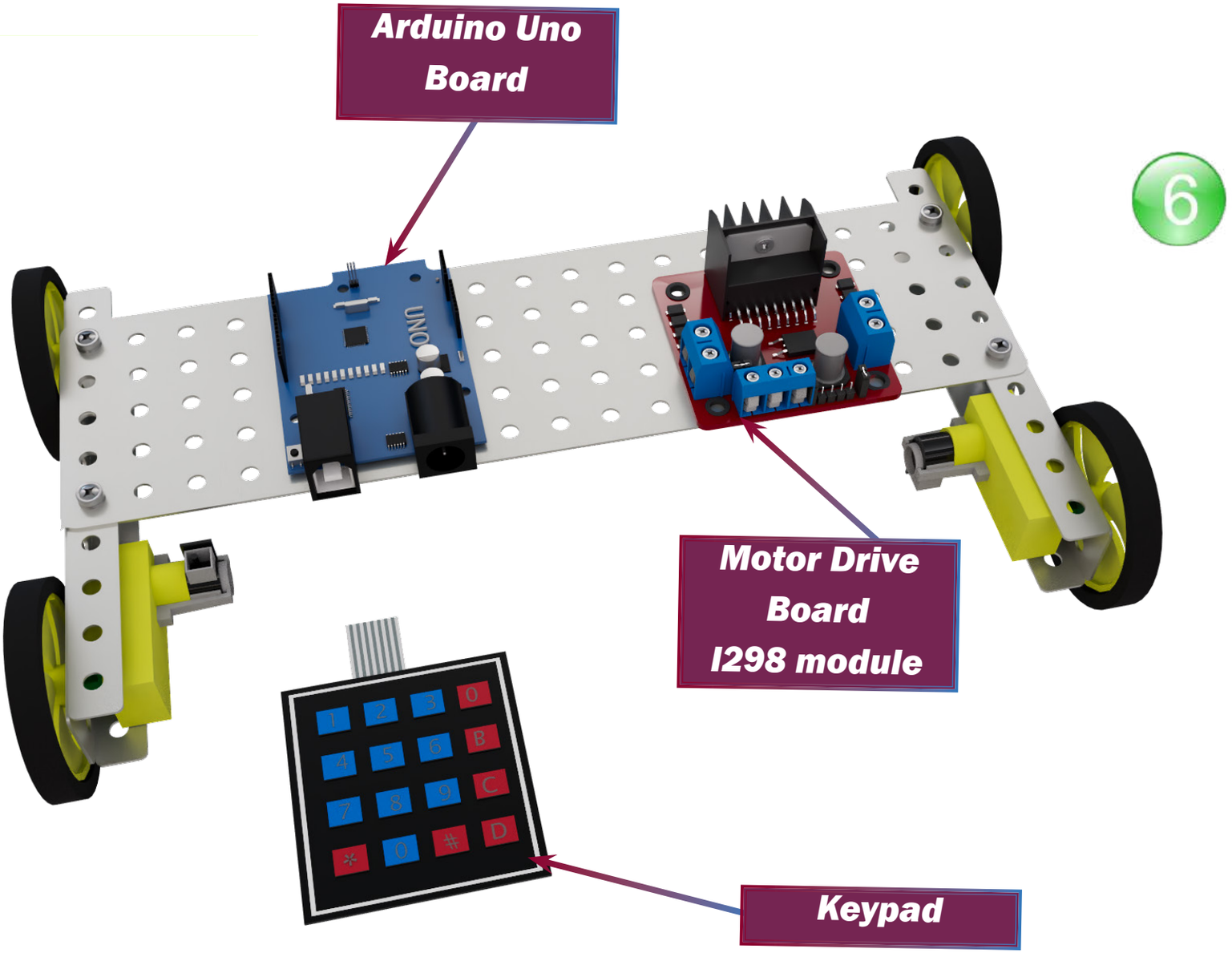
MECHANZ™

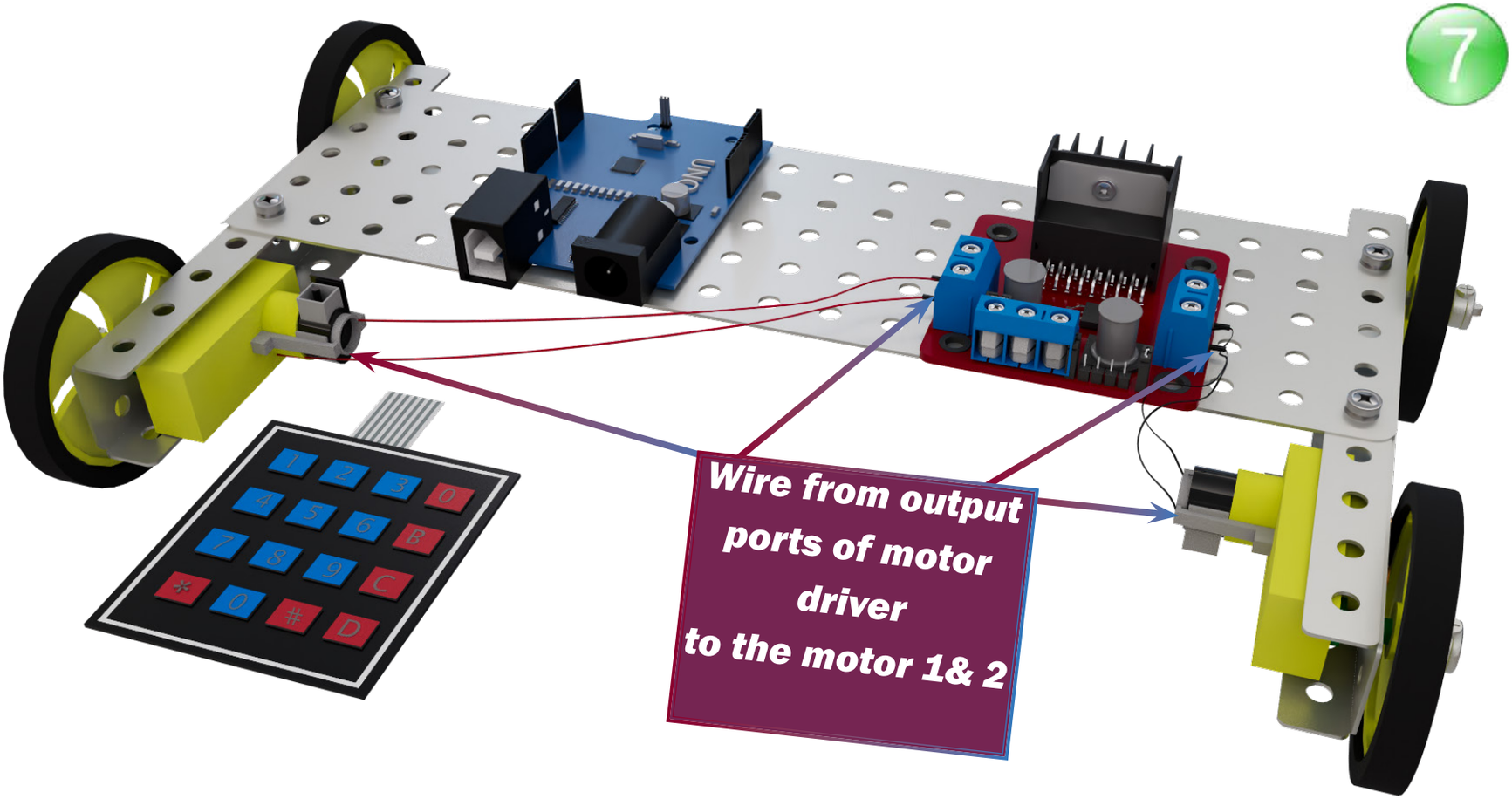


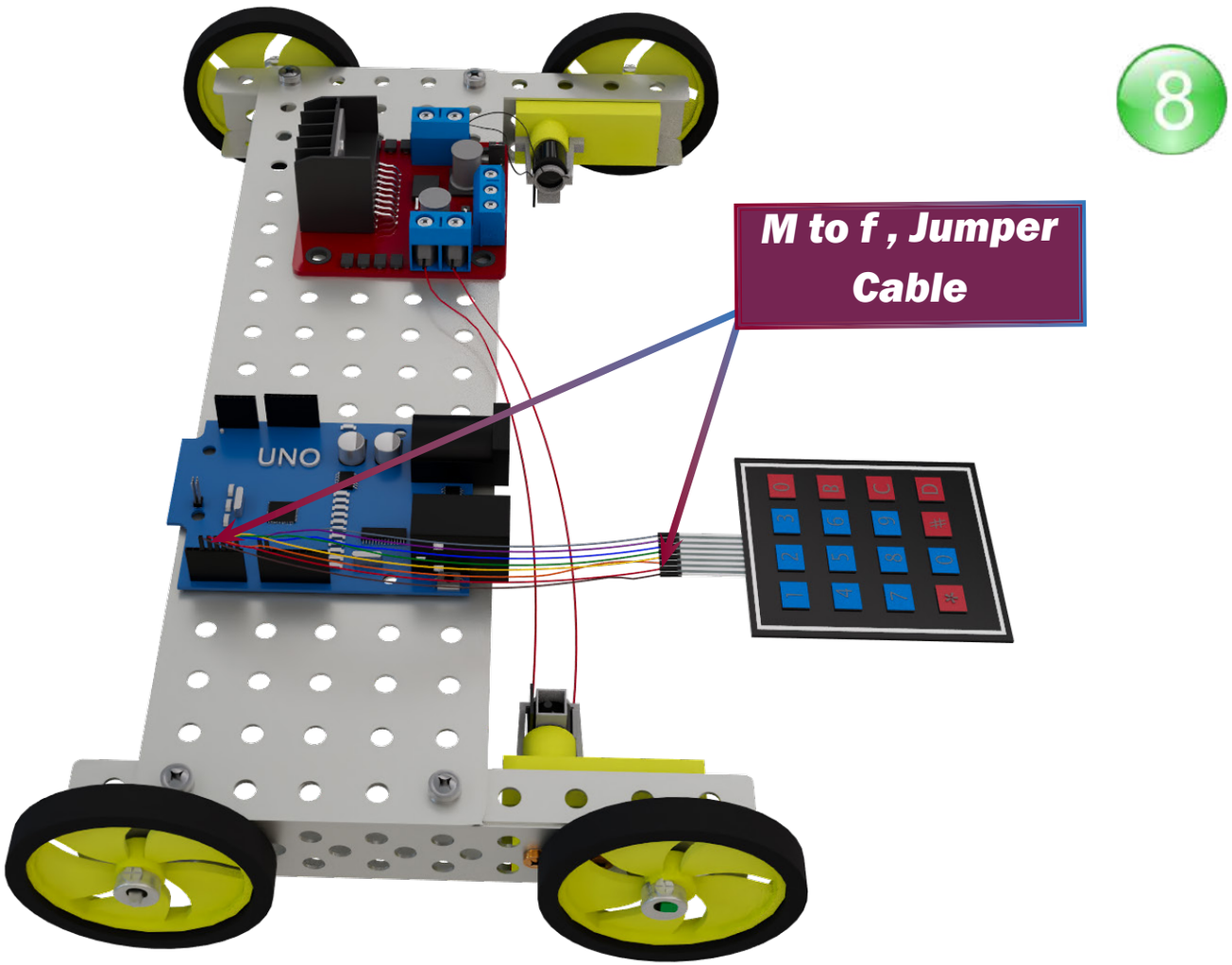


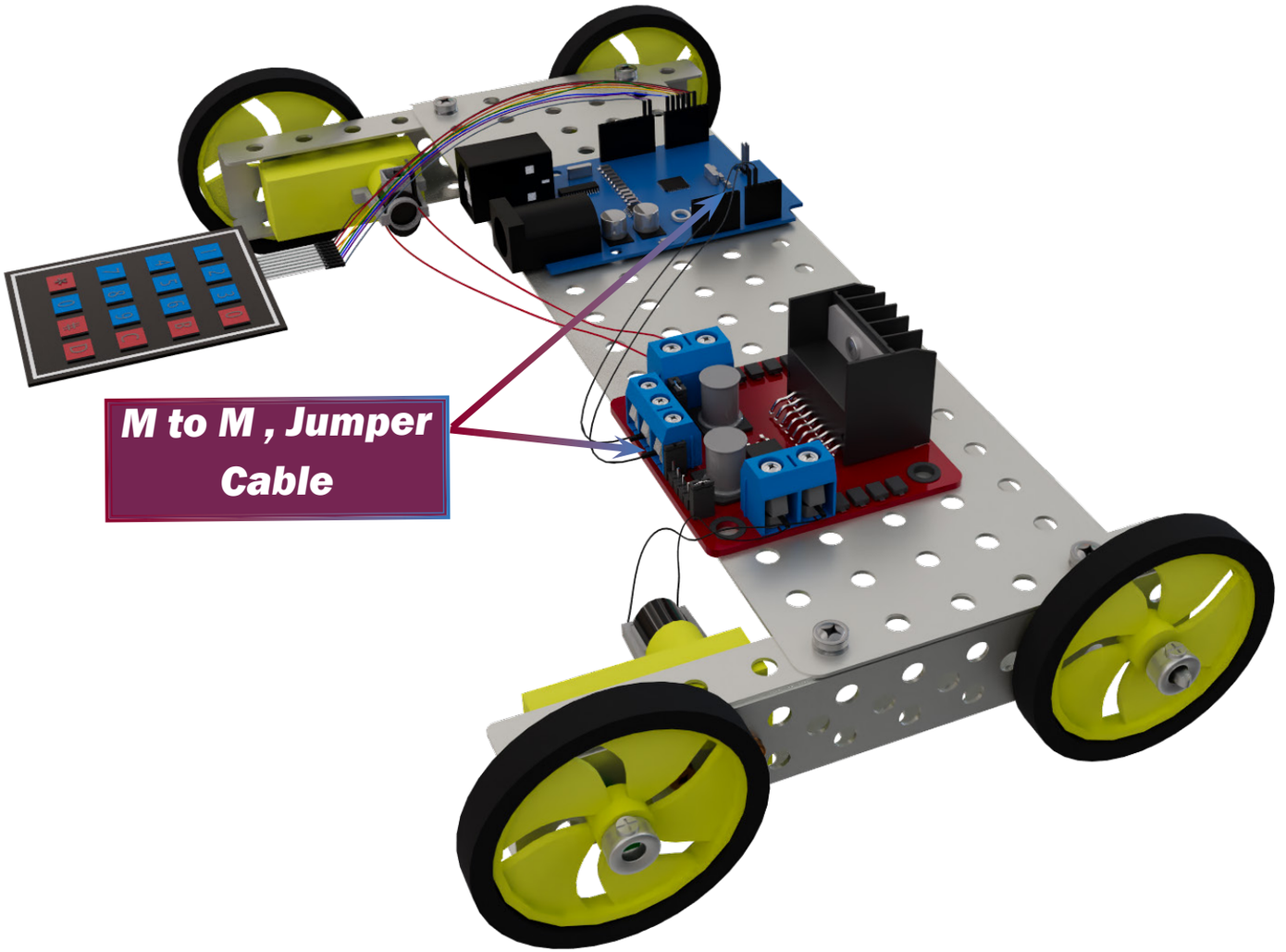
MECHANZ™



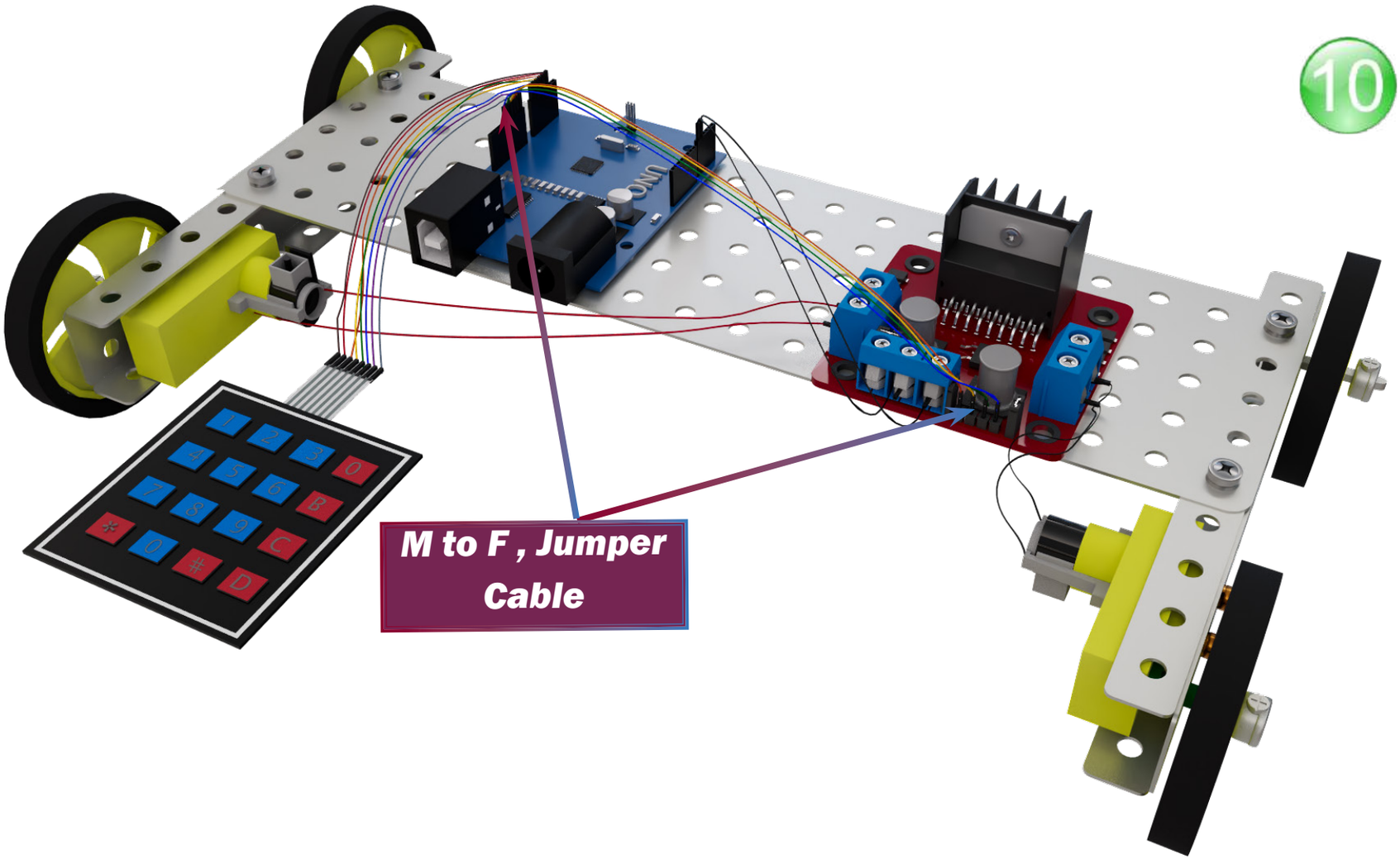






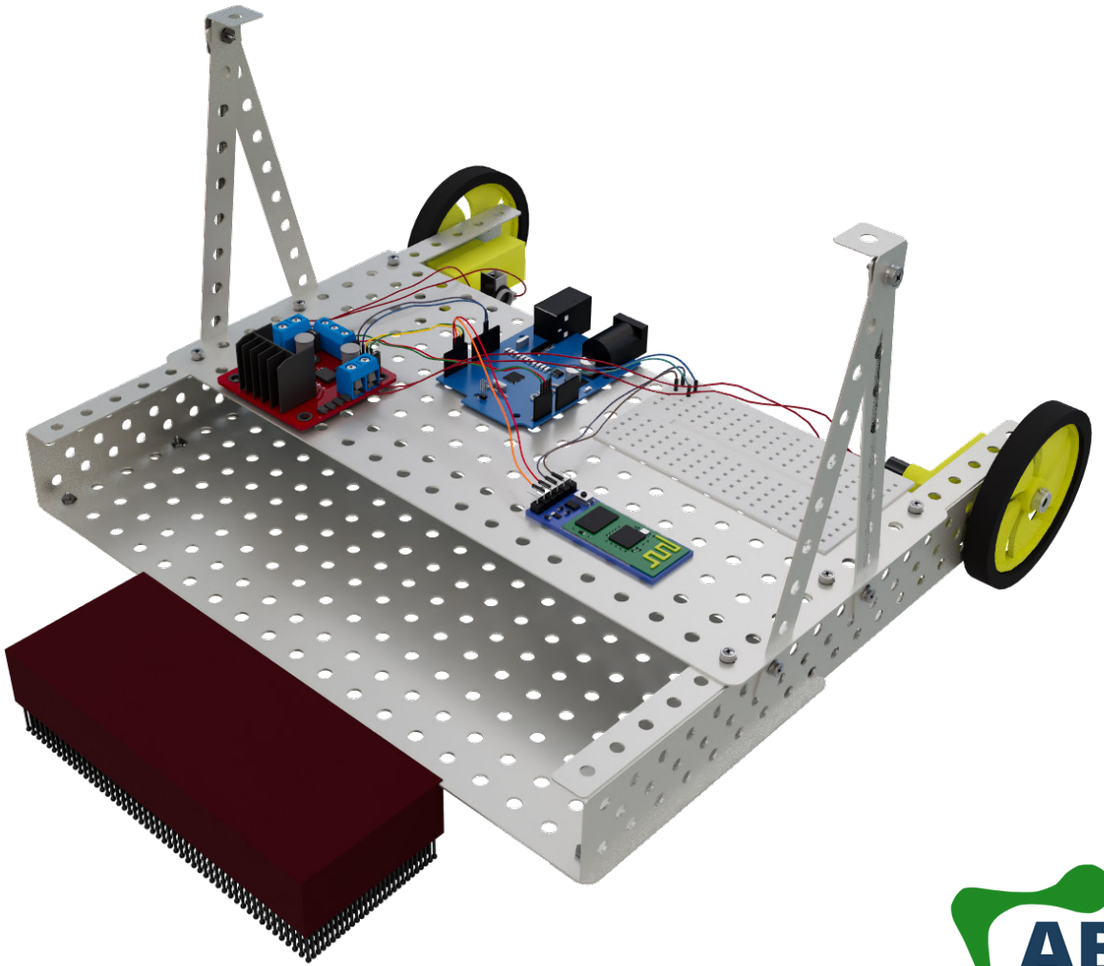


9



MECHANZ™

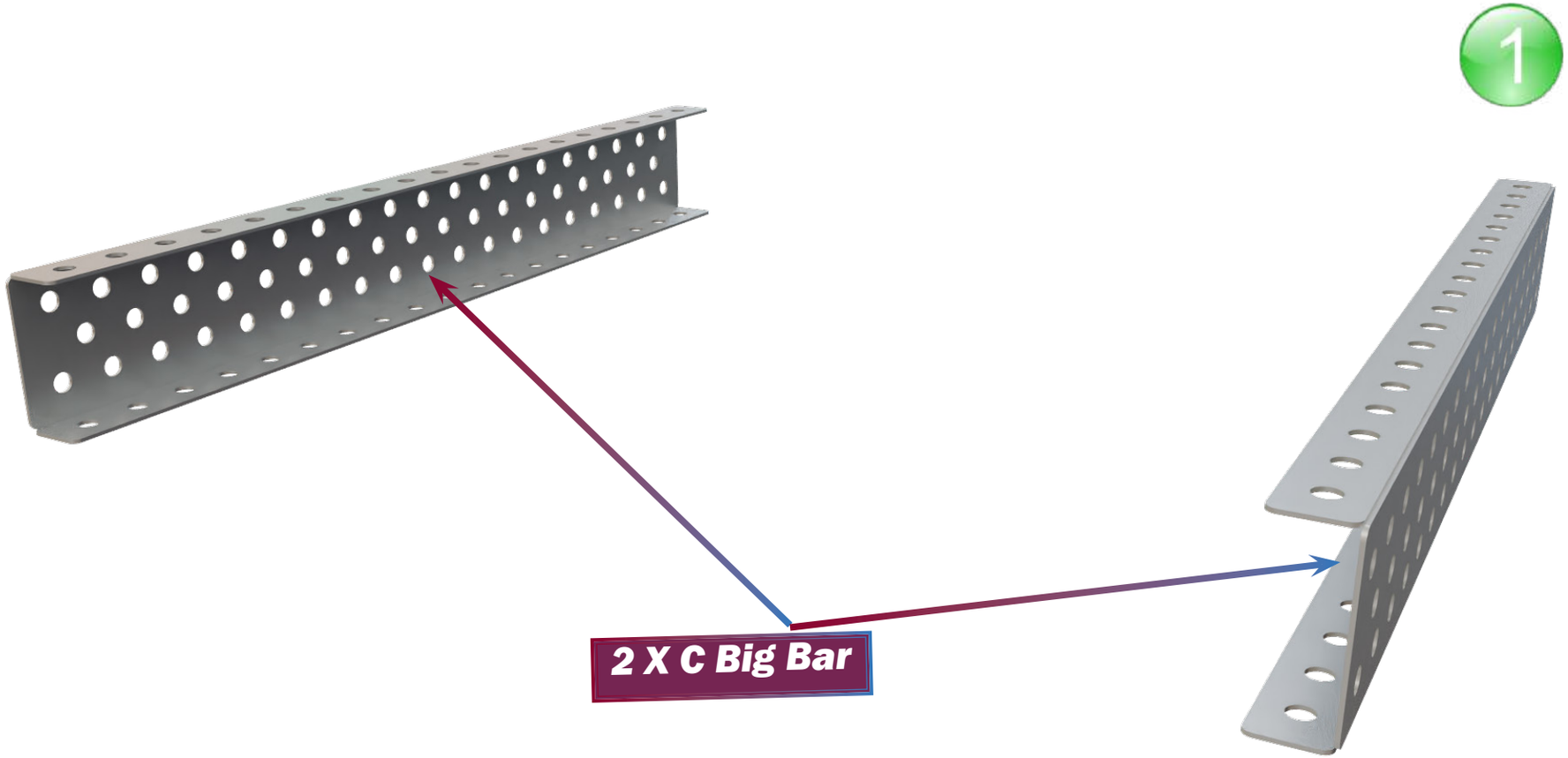
Cleaning Bot



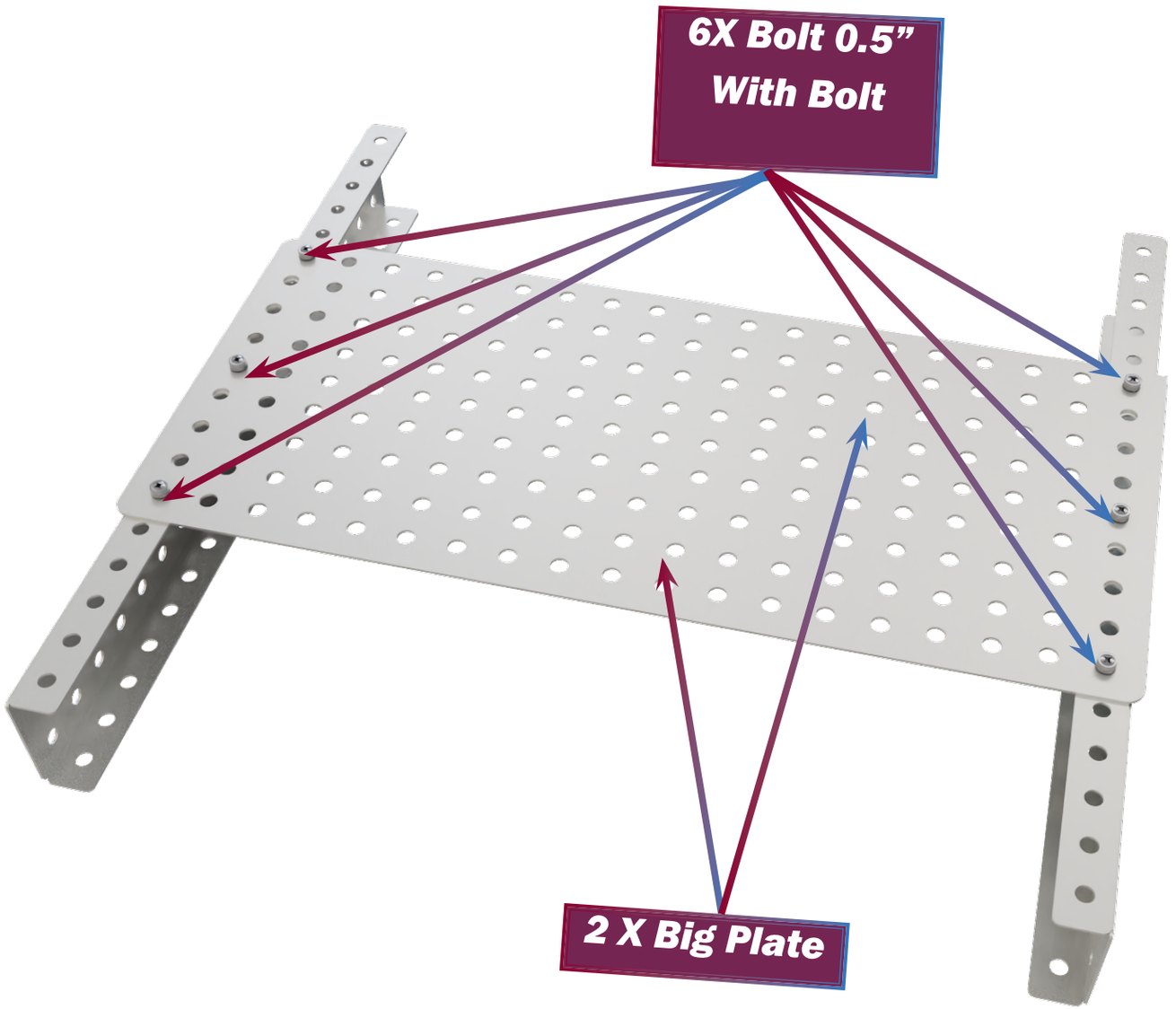
Product of

ABLEDUCATION™

THINK • EXPLORE • CREATE • INNOVATE

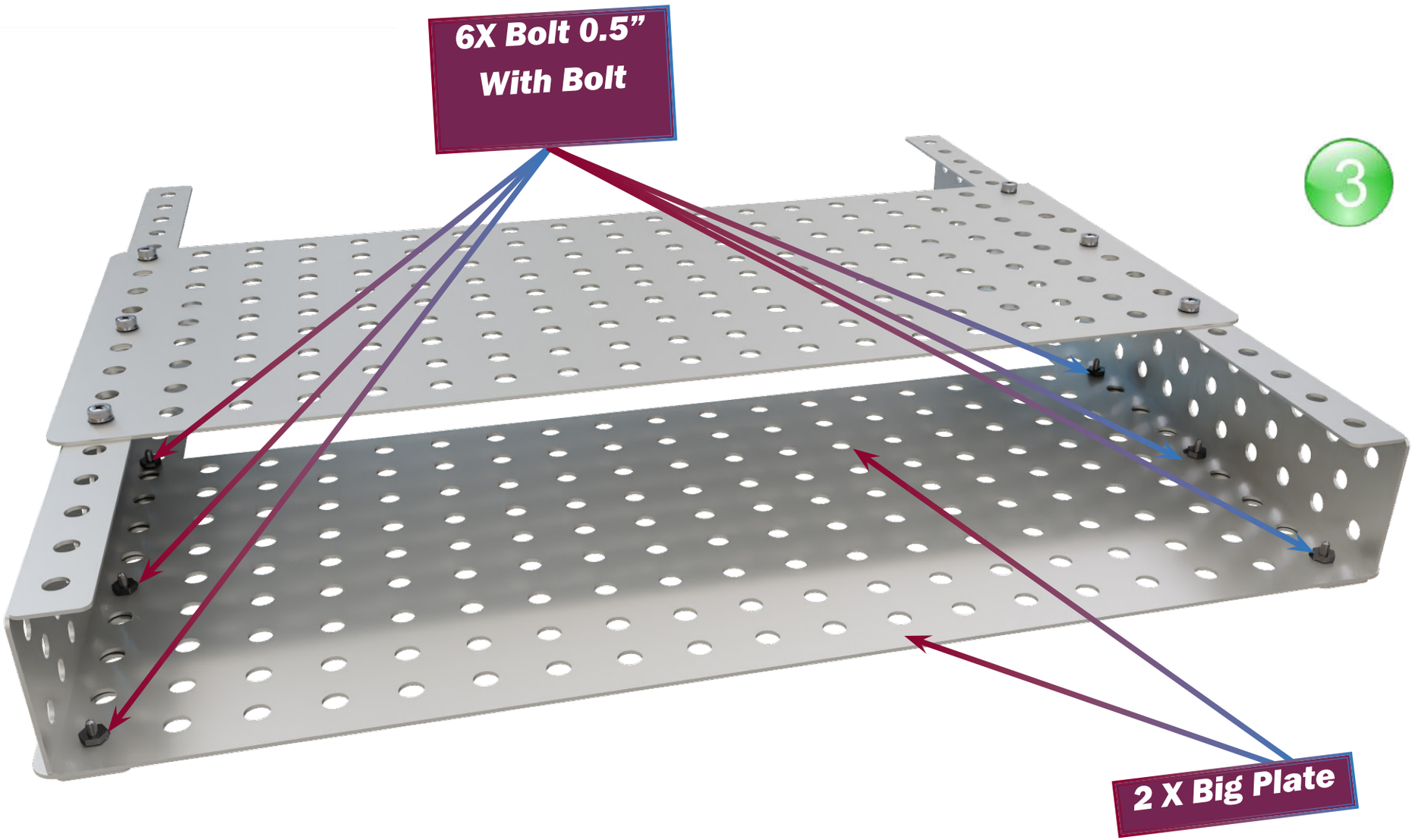


MECHANZ™

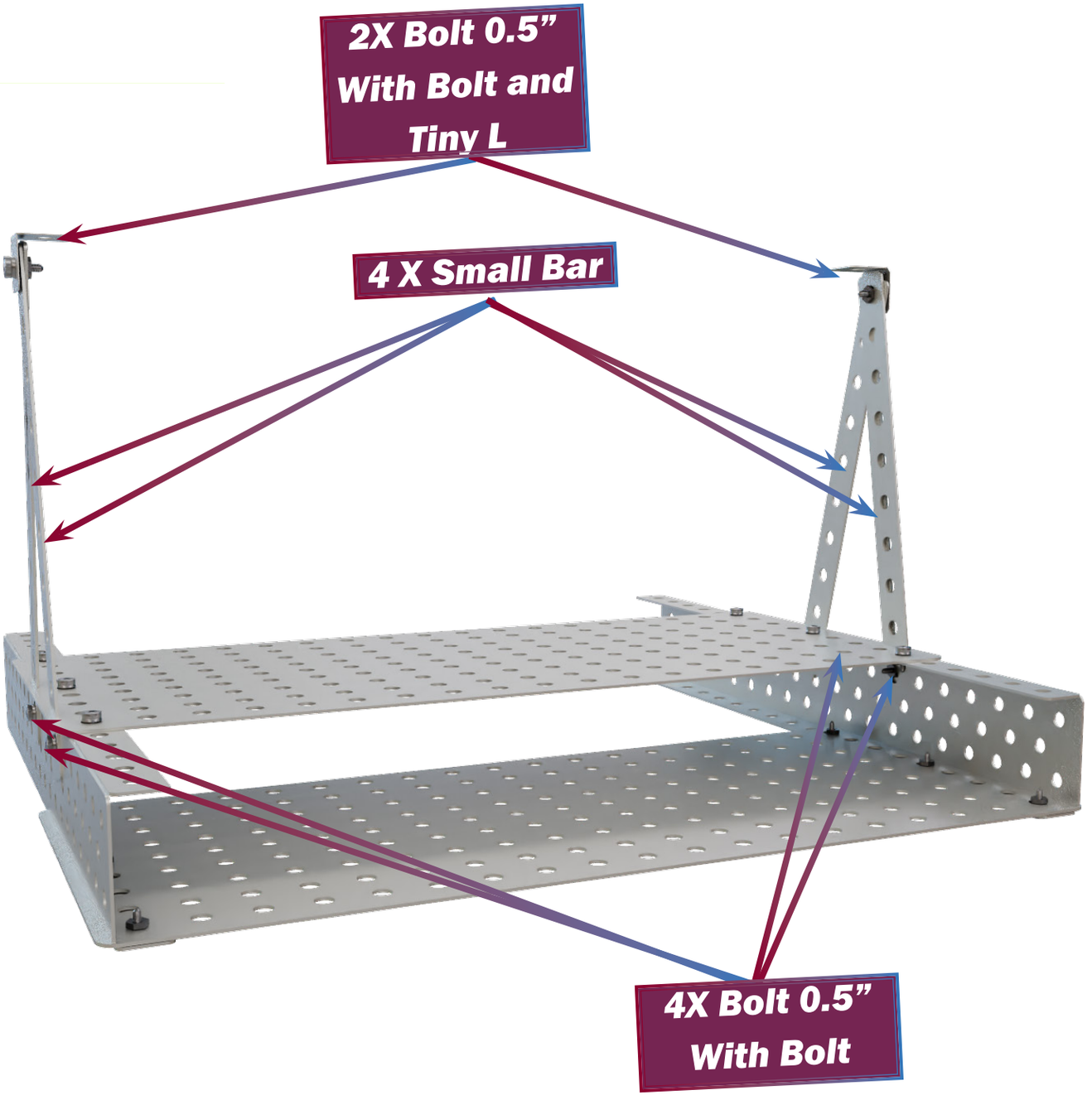


2

MECHANZ™

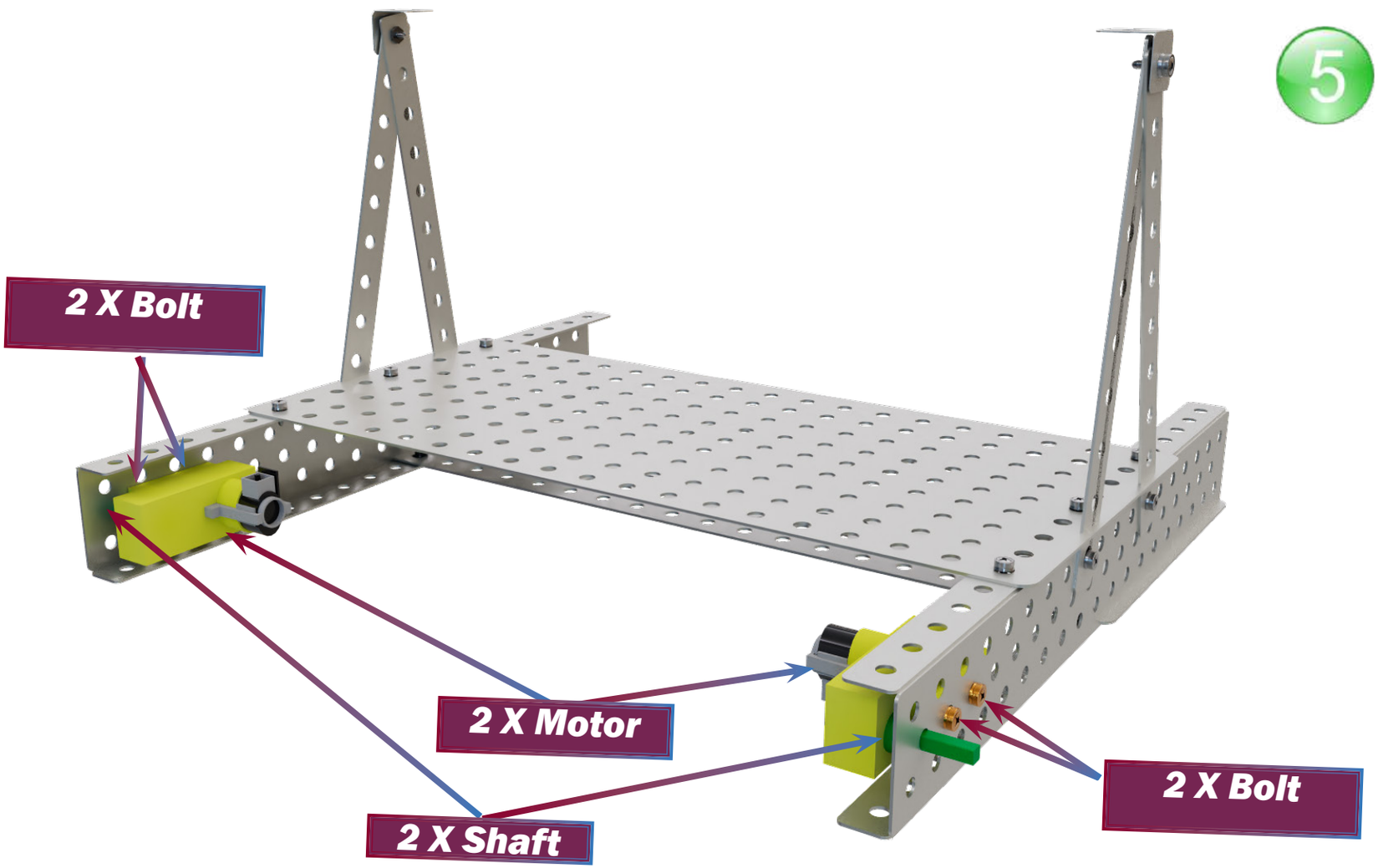


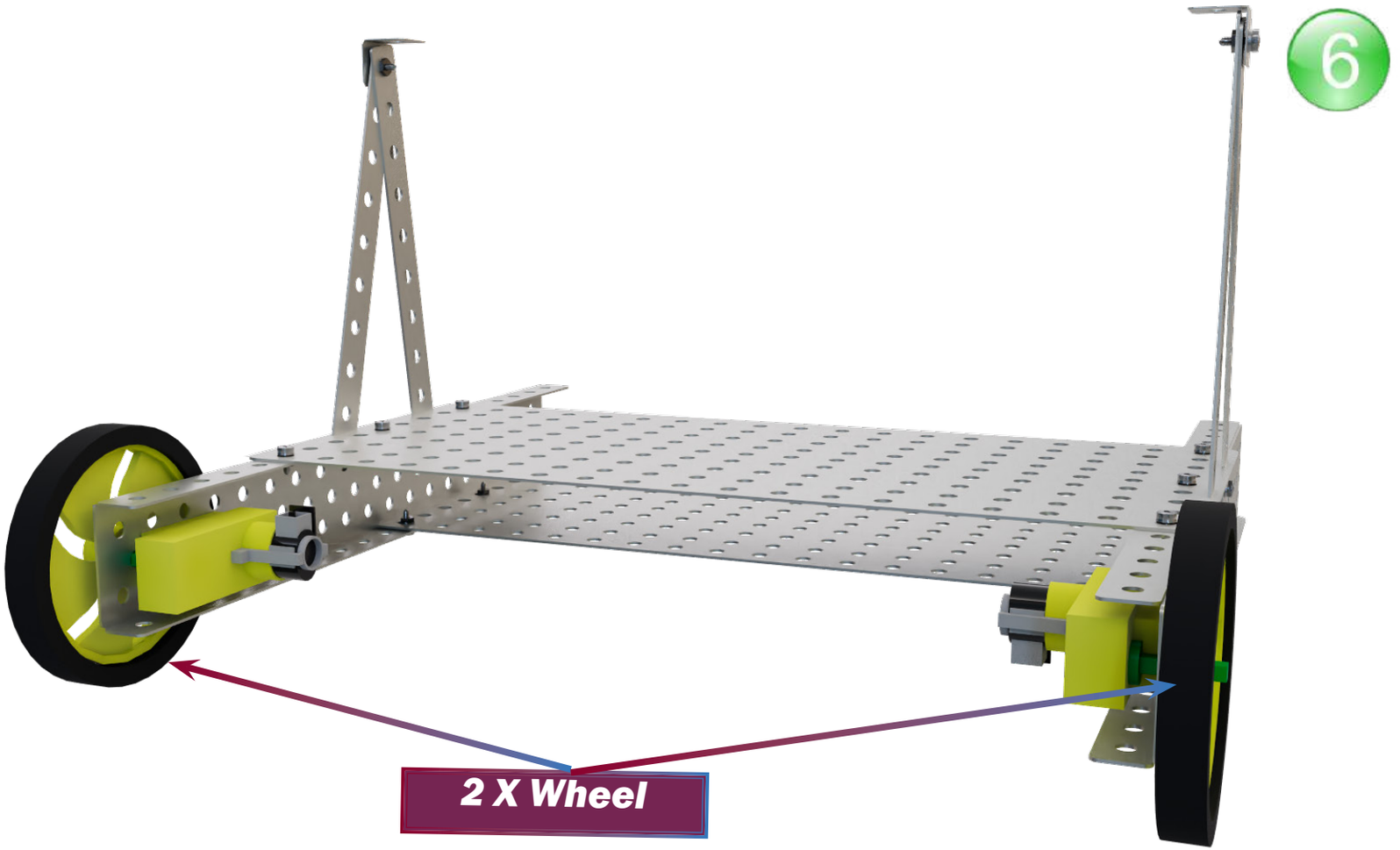
MECHANZ™



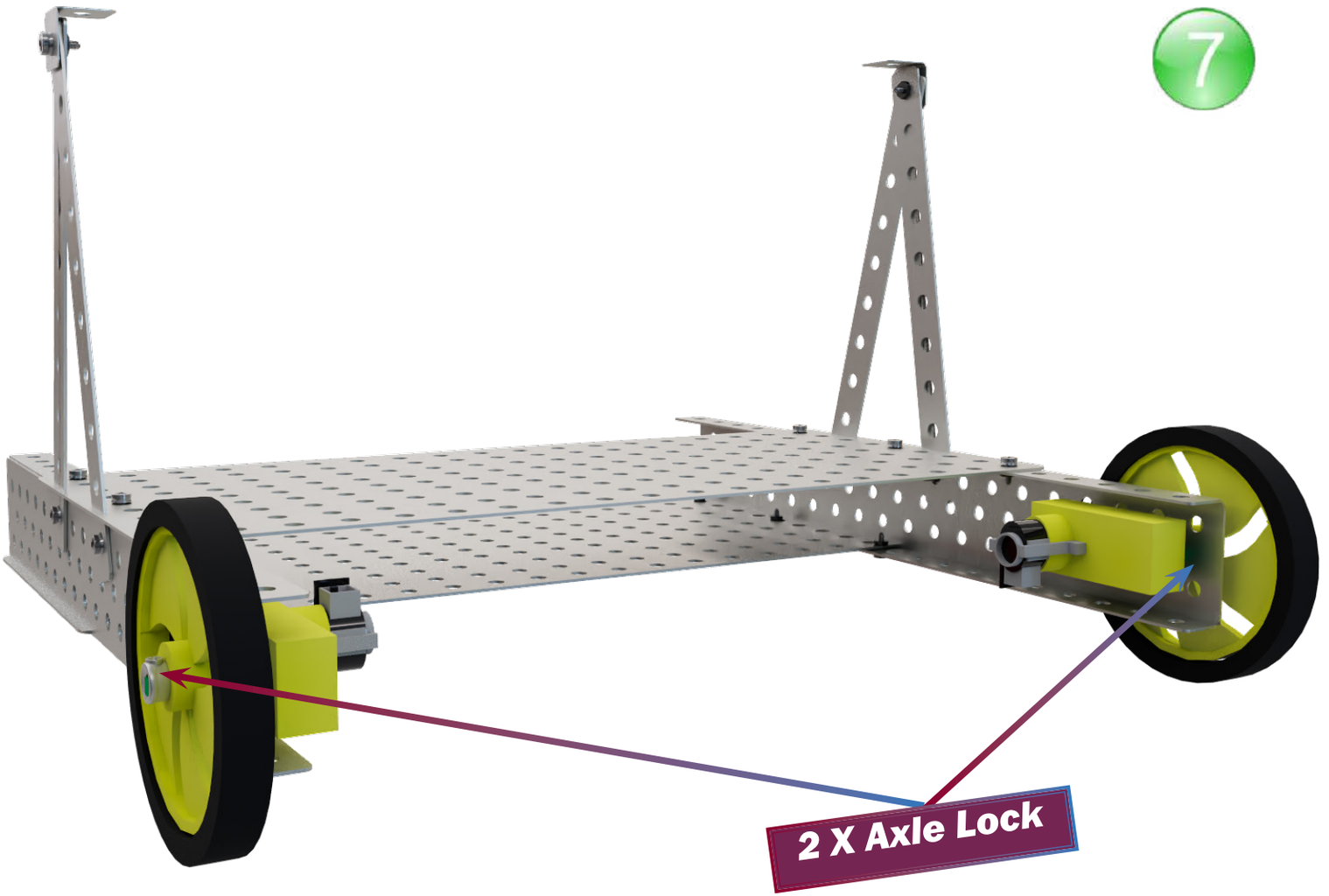
4

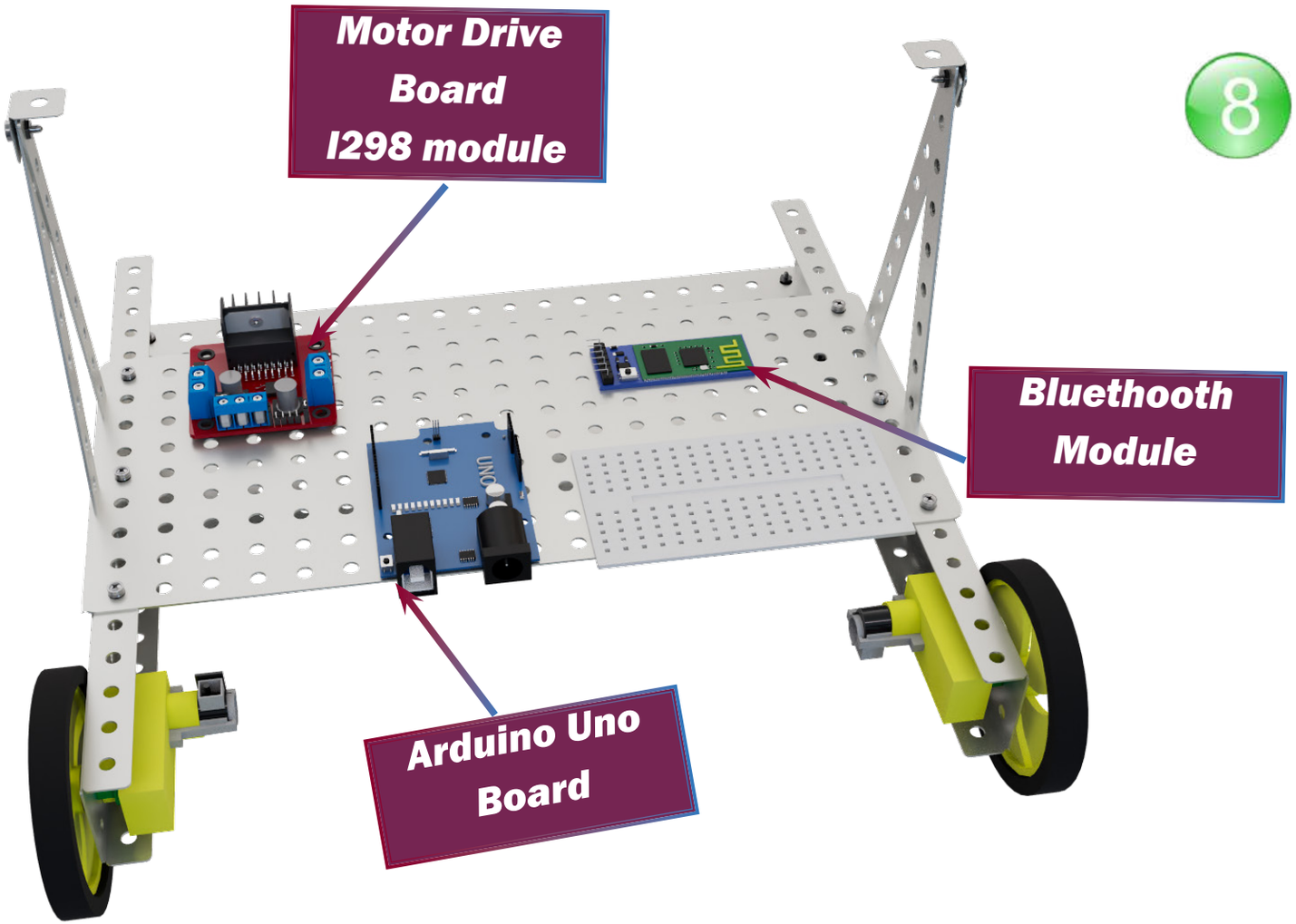
MECHANZ™



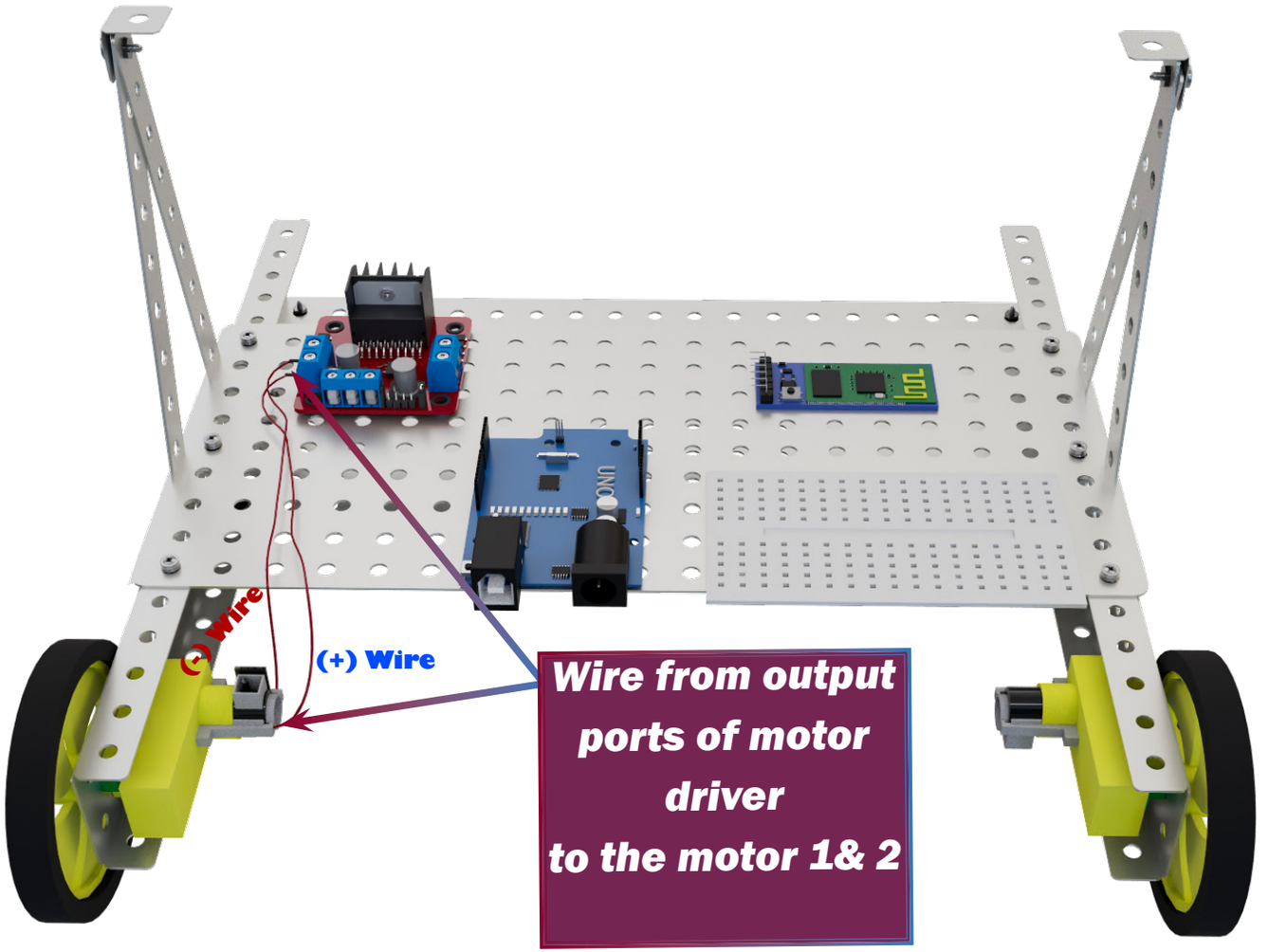


MECHANZ™

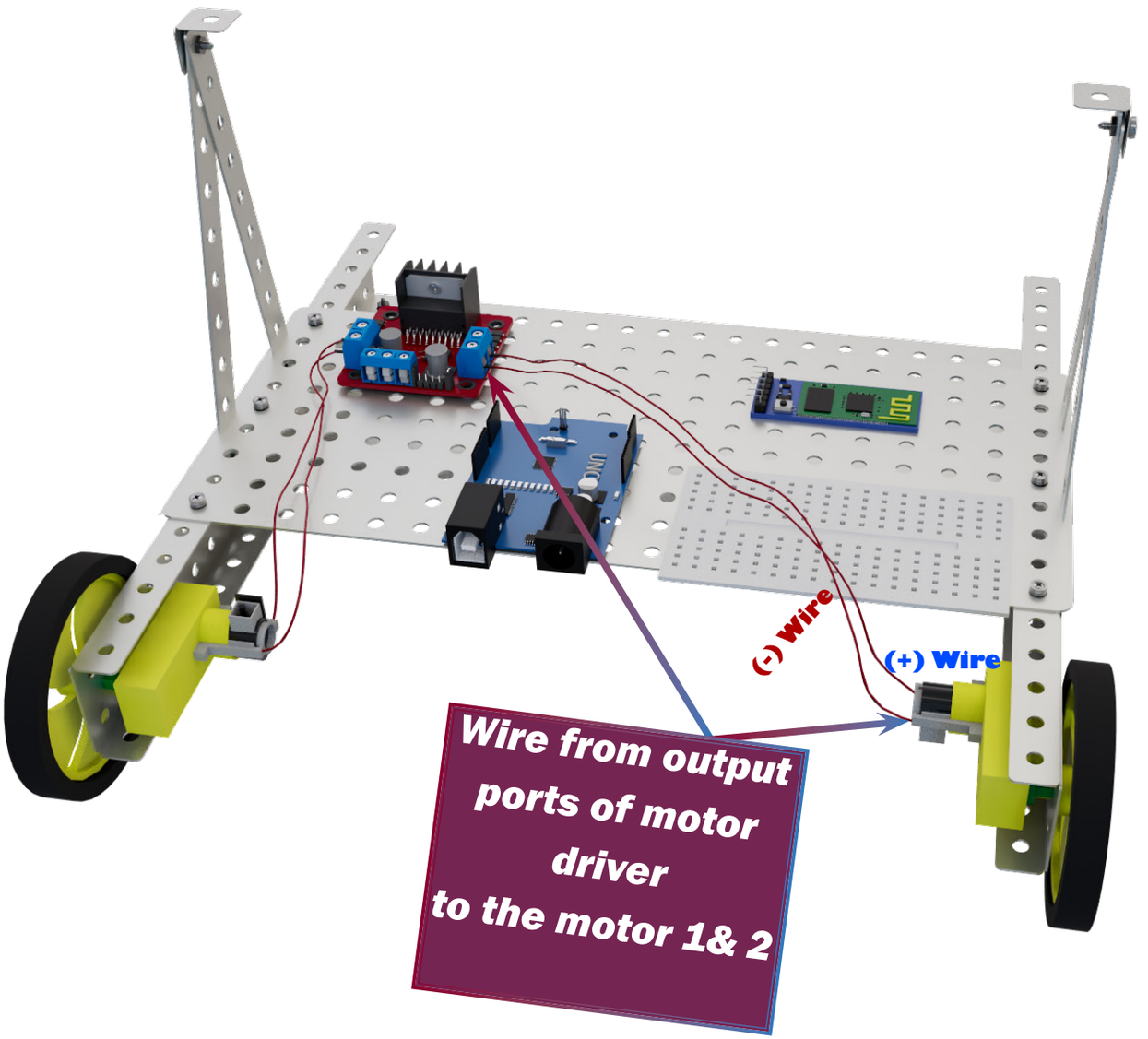




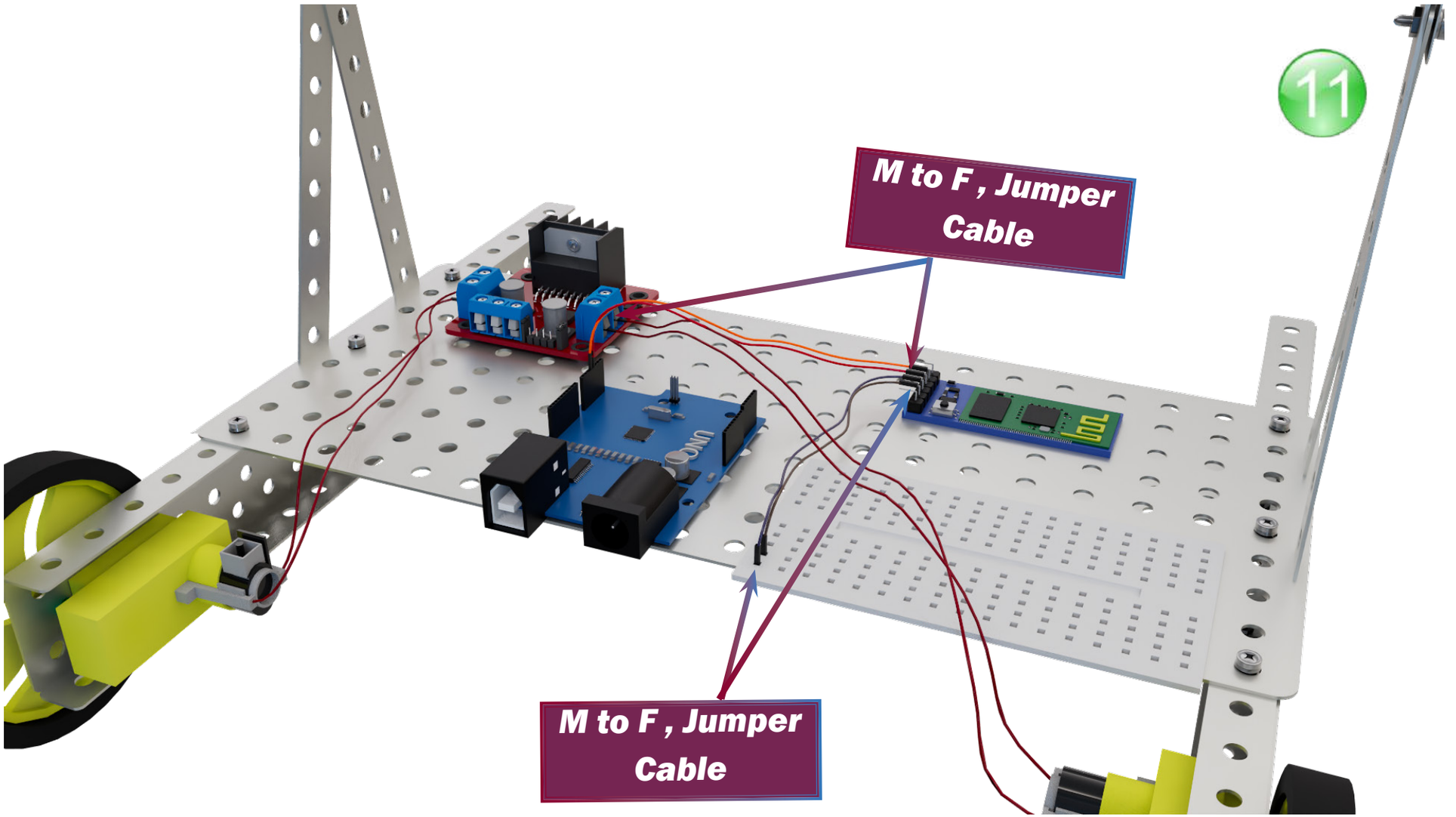
9



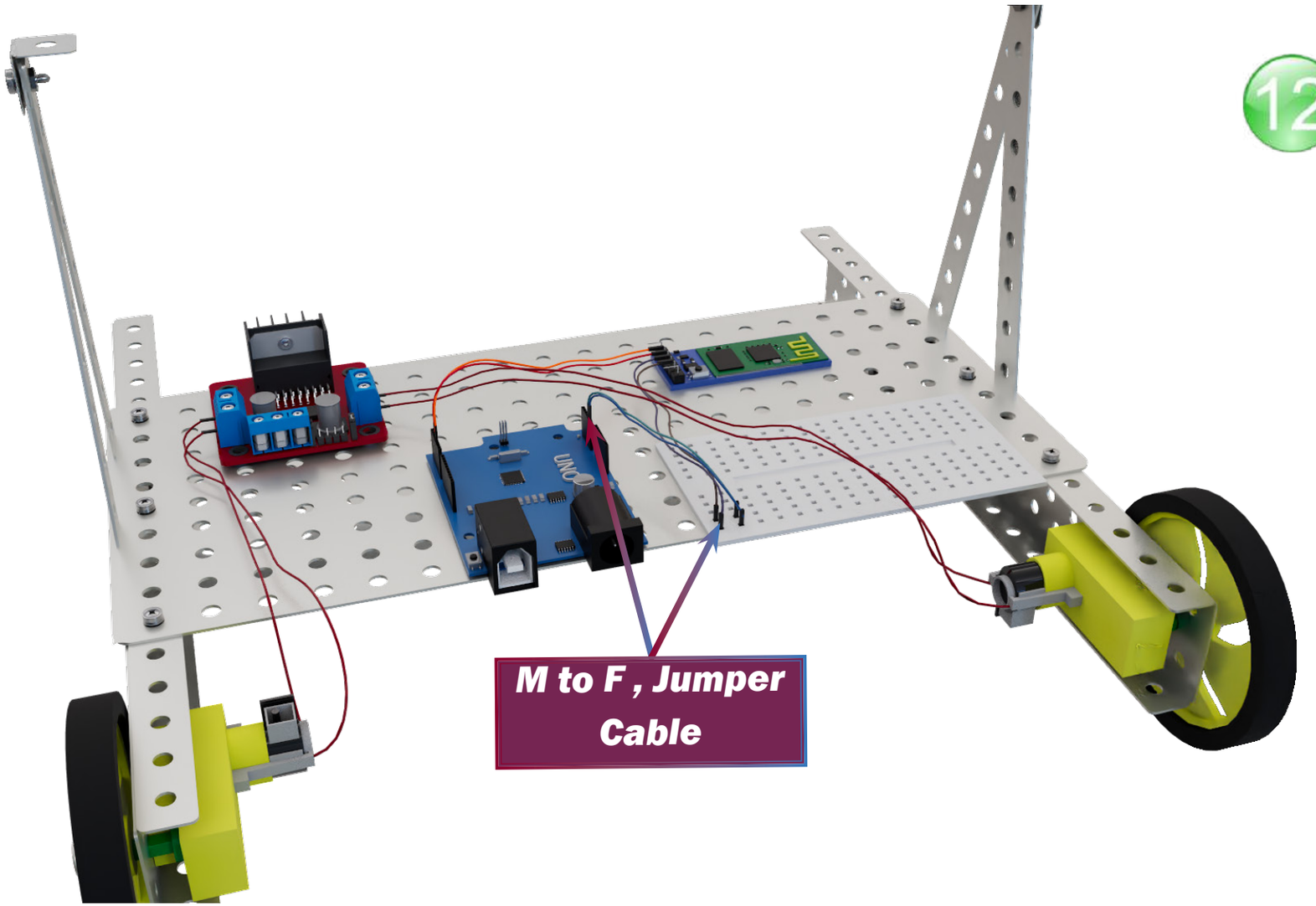
10

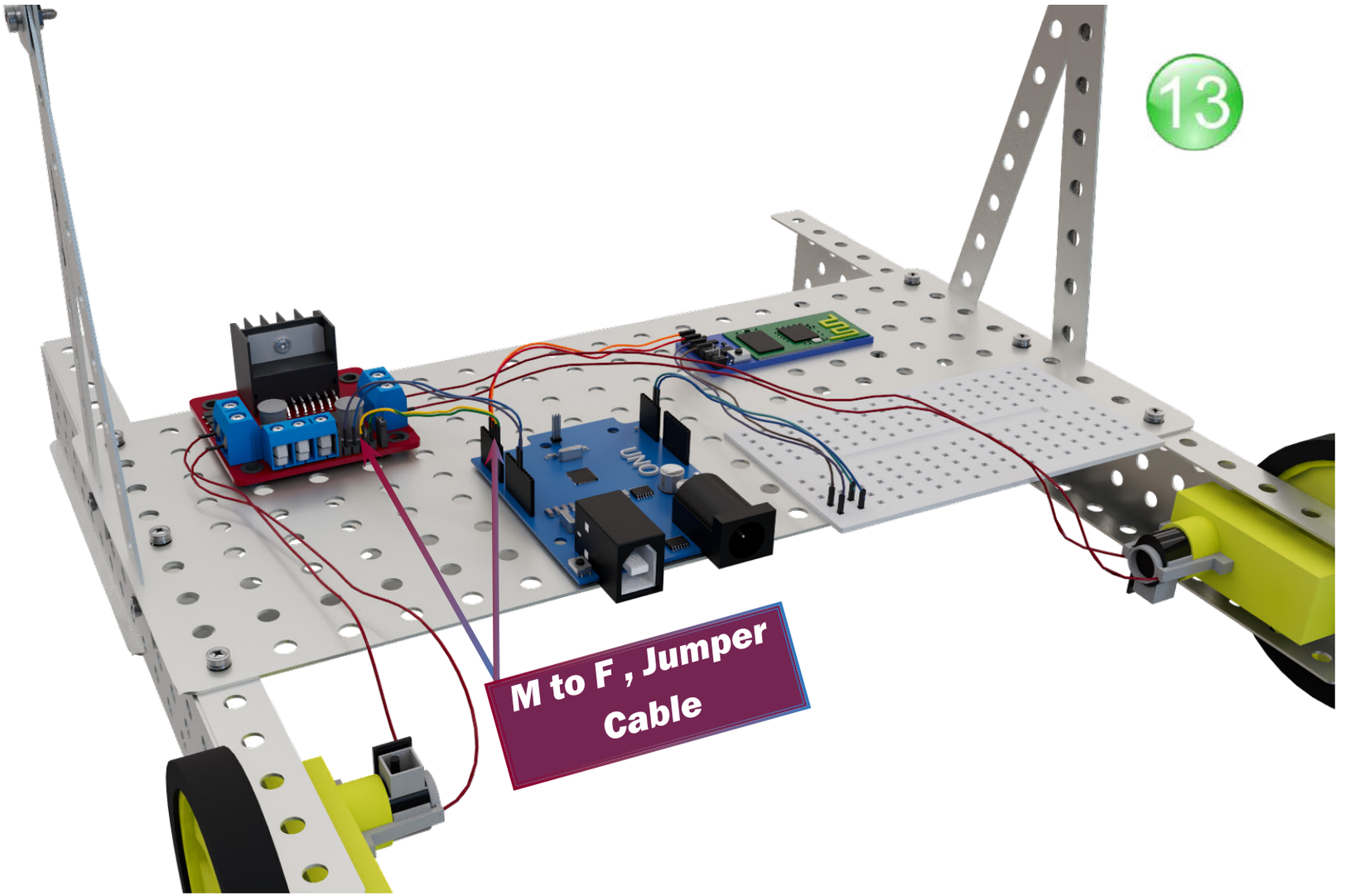

(-) Wire
(+) Wire
Wire from output ports of motor driver to the motor 1& 2

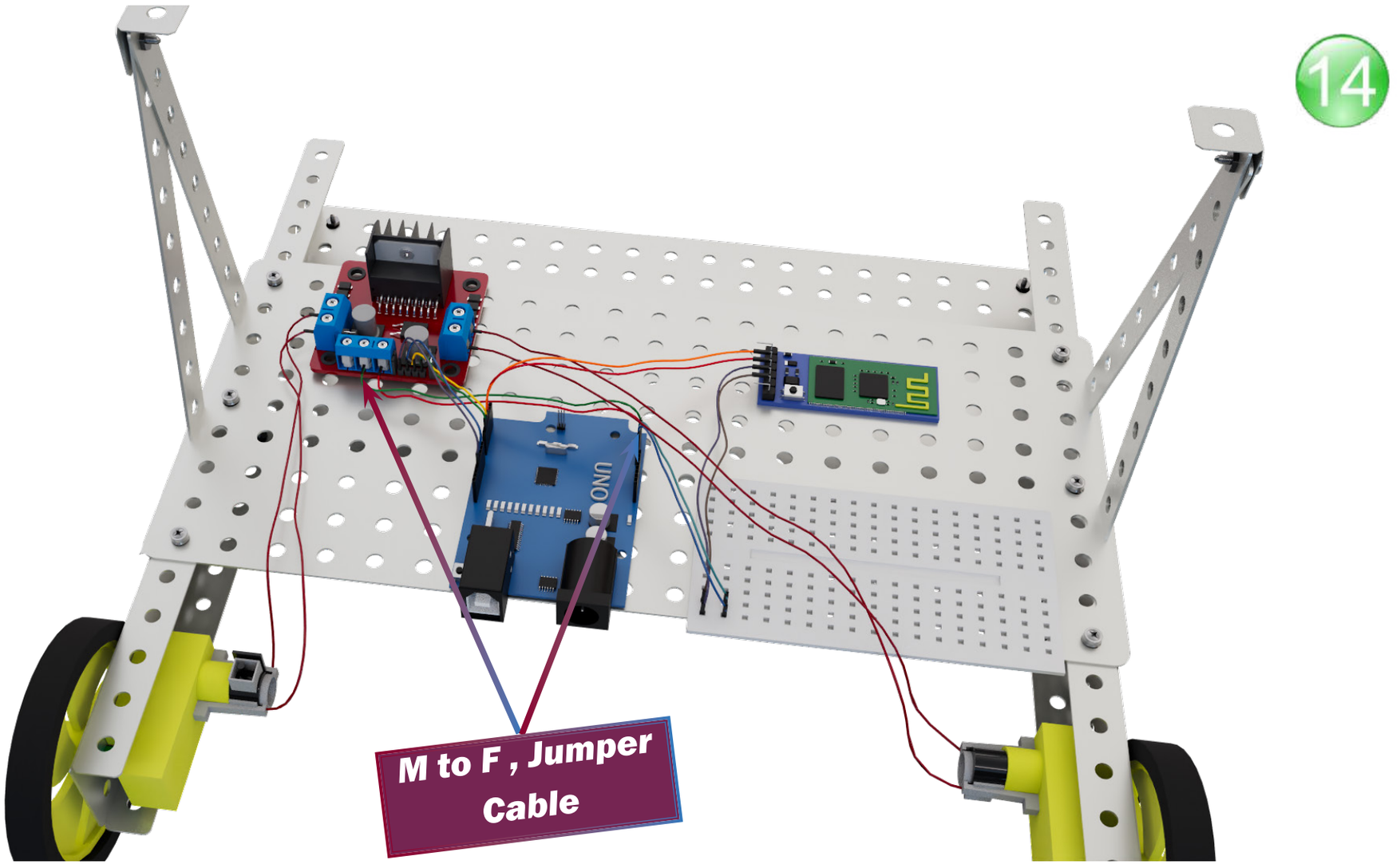
MECHANZ™



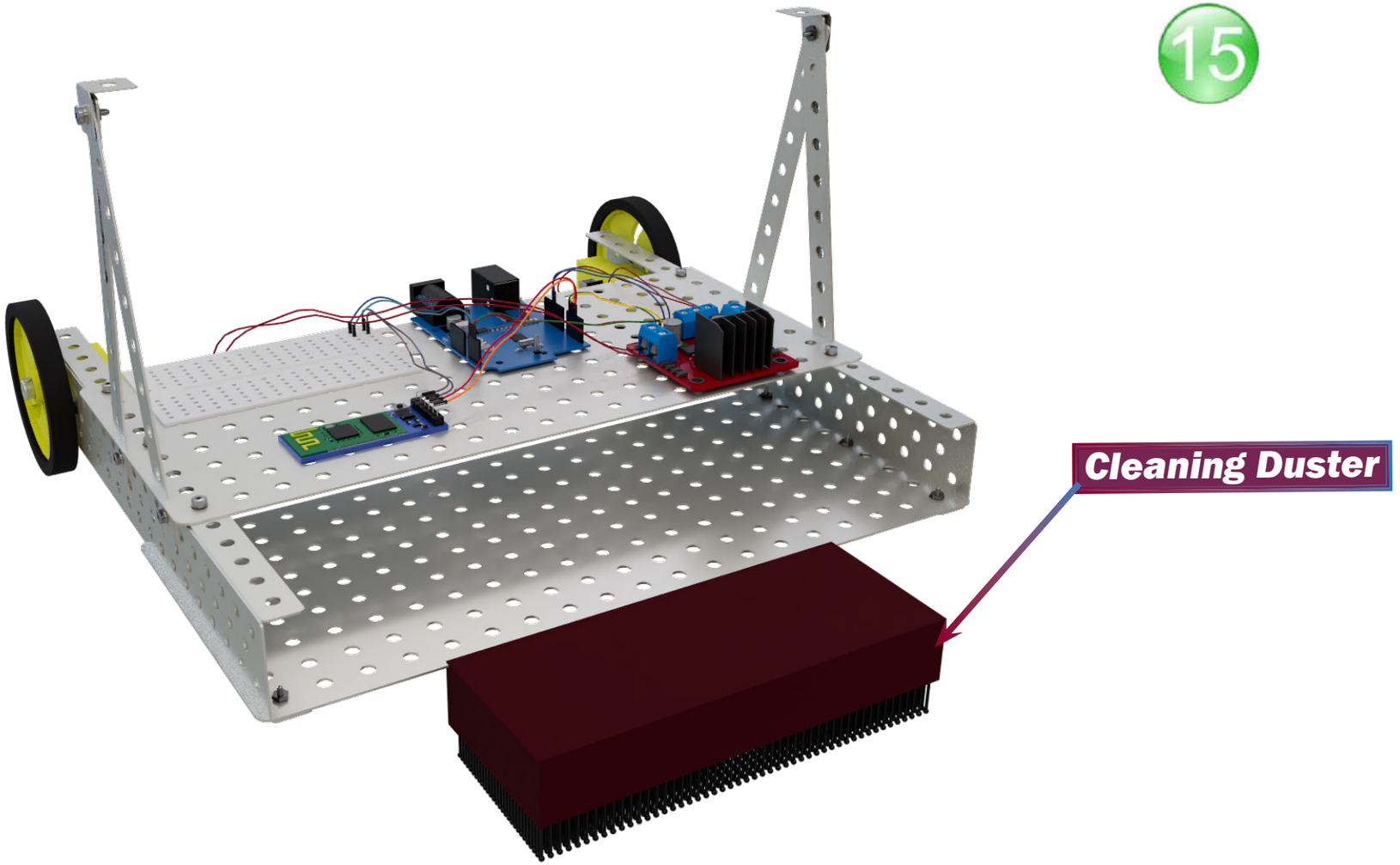
MECHANZ™







15



Coding For Cleaning Bot

```
String readvoice;
int k=0;
void setup() {
Serial.begin(9600);
pinMode(2,OUTPUT);
pinMode(3,OUTPUT);
pinMode(4,OUTPUT);
pinMode(5,OUTPUT);
}

void loop() {
while (Serial.available())
{
delay(3);
char c = Serial.read();
readvoice += c;
}

if(readvoice.length() > 0)
{
Serial.println(readvoice);

if(readvoice == “forward”)
```

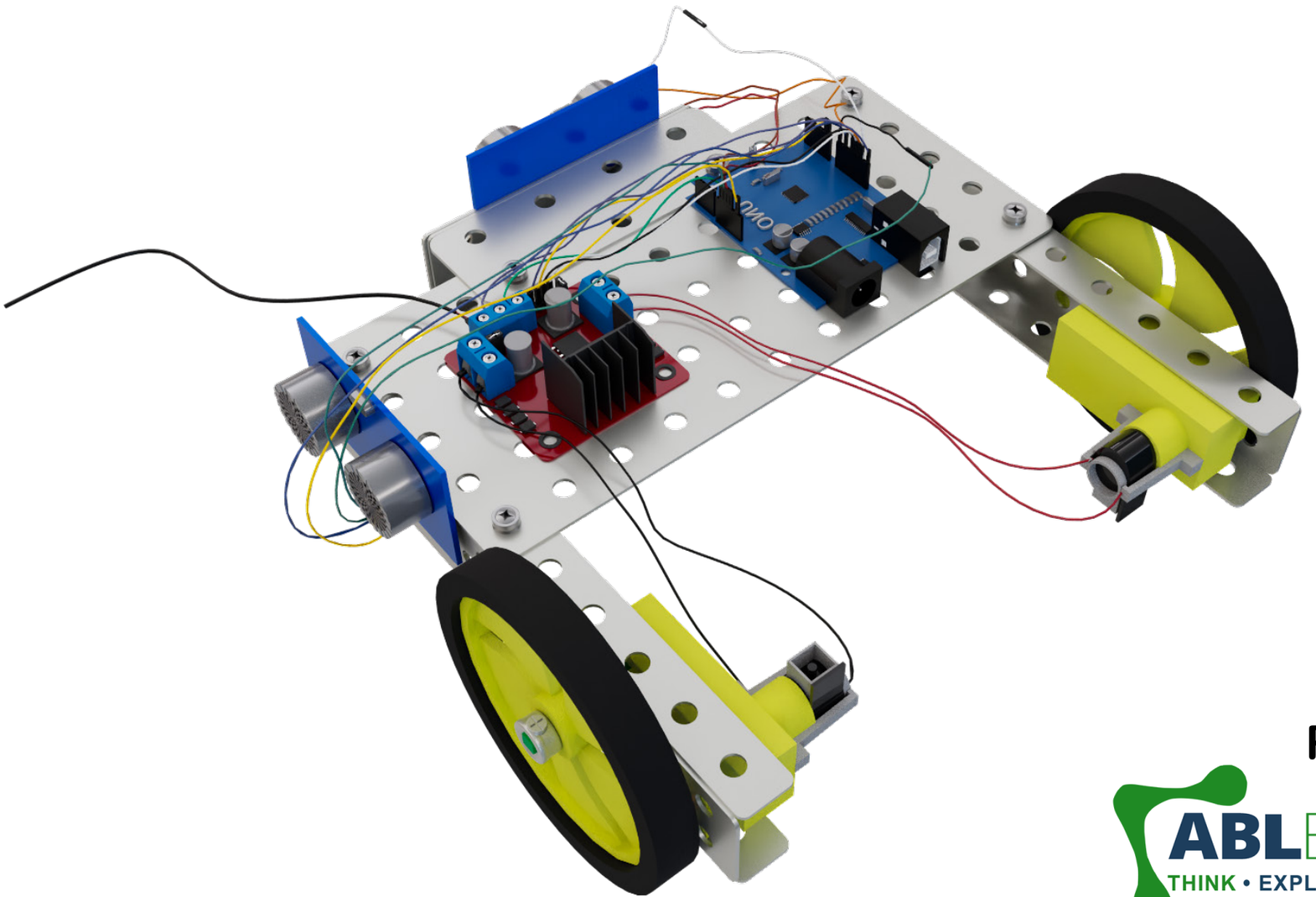
```
{
digitalWrite(2, HIGH);
digitalWrite(3, LOW);
digitalWrite(4, HIGH);
digitalWrite(5, LOW);
k=1;
}

if(readvoice == “backward”)
{
digitalWrite(2, LOW);
digitalWrite(3, HIGH);
digitalWrite(4, LOW);
digitalWrite(5, HIGH);
k=2;
}

if(readvoice == “right”)
{
if (k==2)
{
digitalWrite(2, HIGH);
digitalWrite(3, LOW);
digitalWrite(4, LOW);
digitalWrite(5, LOW);
delay(1000);
digitalWrite(2, LOW);
```

```
digitalWrite(3, HIGH);
digitalWrite(4, LOW);
digitalWrite(5, HIGH);
}
else
{
digitalWrite(2, HIGH);
digitalWrite(3, LOW);
digitalWrite(4, LOW);
digitalWrite(5, LOW);
delay(1000);
digitalWrite(2, HIGH);
digitalWrite(3, LOW);
digitalWrite(4, HIGH);
digitalWrite(5, LOW);
}
}

if(readvoice == “left”)
{
if (k==2)
{
digitalWrite(2, LOW);
digitalWrite(3, LOW);
digitalWrite(4, HIGH);
digitalWrite(5, LOW);
delay(1000);
digitalWrite(2, LOW);
digitalWrite(3, HIGH);
```

```
digitalWrite(4, LOW);
digitalWrite(5, HIGH);
}
else
{
digitalWrite(2, LOW);
digitalWrite(3, LOW);
digitalWrite(4, HIGH);
digitalWrite(5, LOW);
delay(1000);
digitalWrite(2, HIGH);
digitalWrite(3, LOW);
digitalWrite(4, HIGH);
digitalWrite(5, LOW);
}
}

if(readvoice == “stop”)
{
digitalWrite(2, LOW);
digitalWrite(3, LOW);
digitalWrite(4, LOW);
digitalWrite(5, LOW);
}
}
readvoice=””;
}
```

MECHANZ™

Edge Detecting Bot

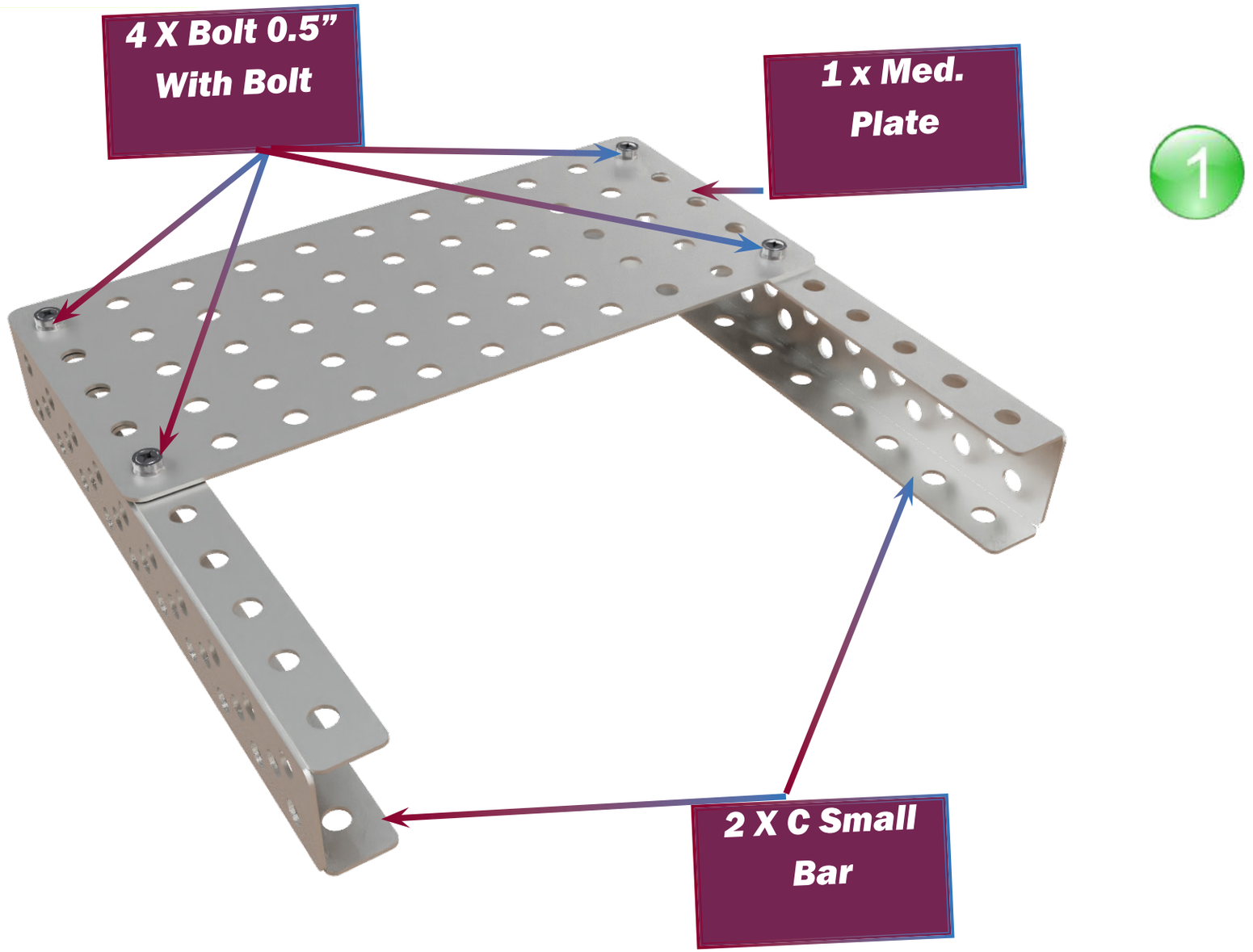


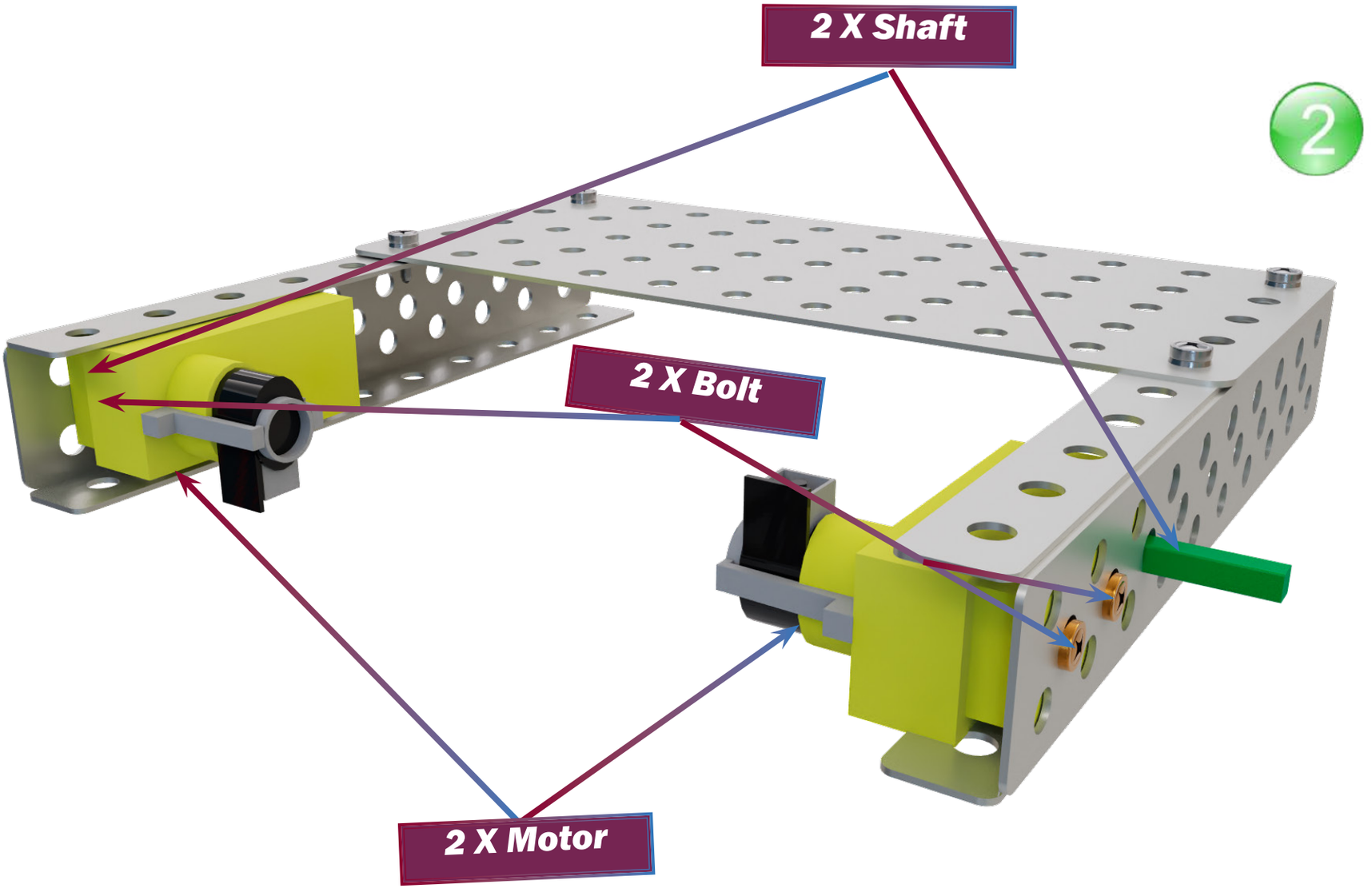
Product of

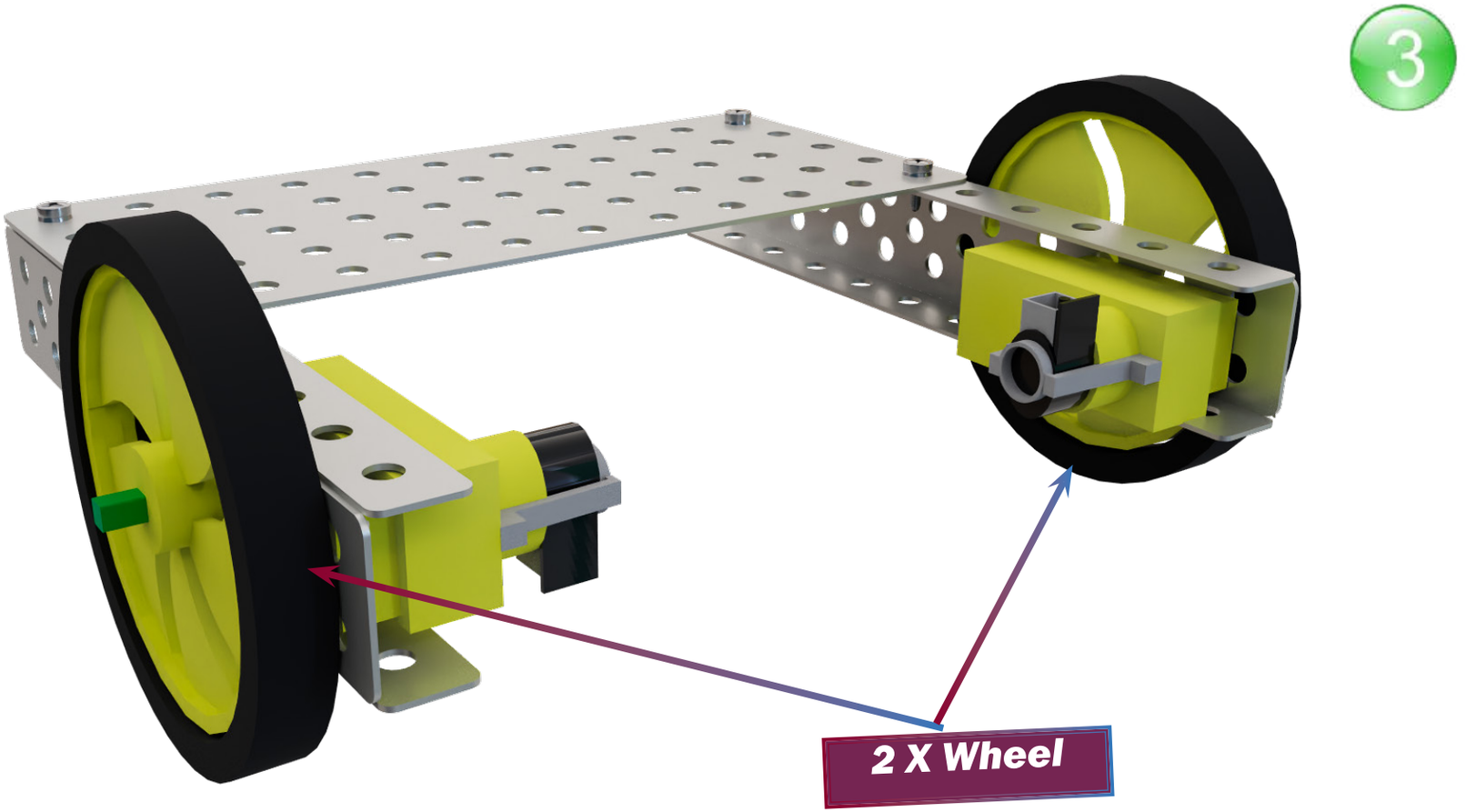
ABLEDUCATION™

THINK • EXPLORE • CREATE • INNOVATE

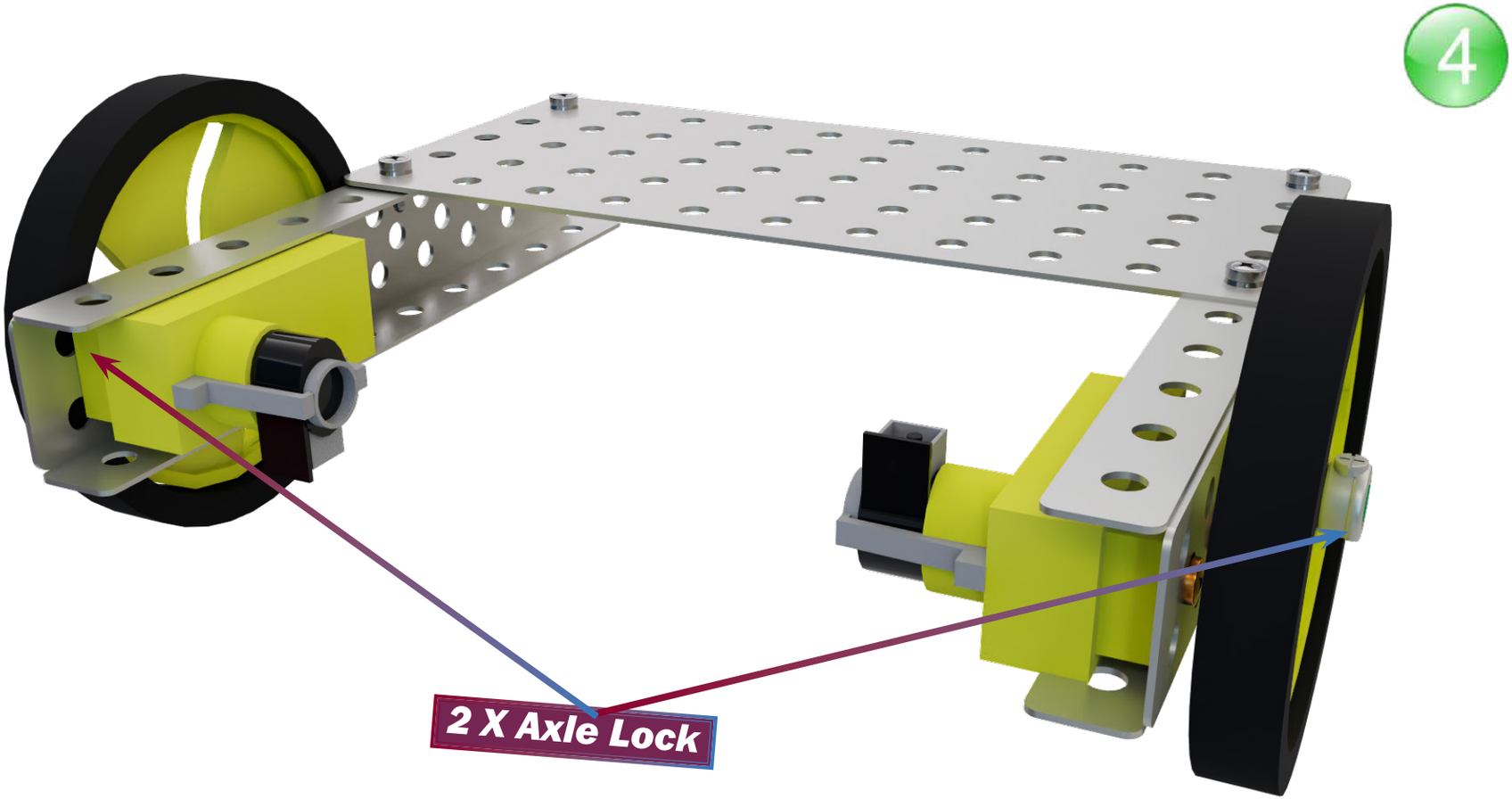
MECHANZ™



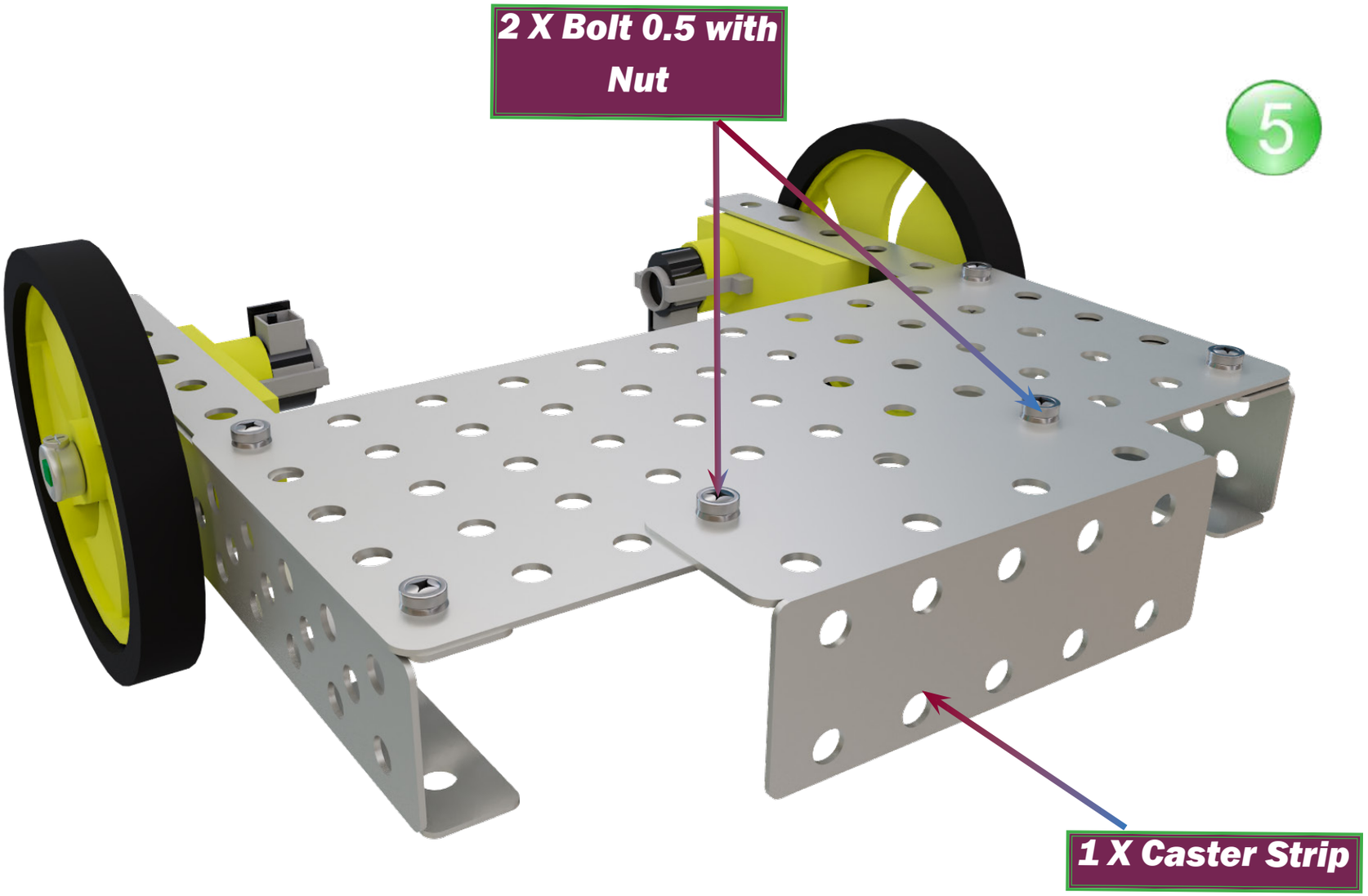


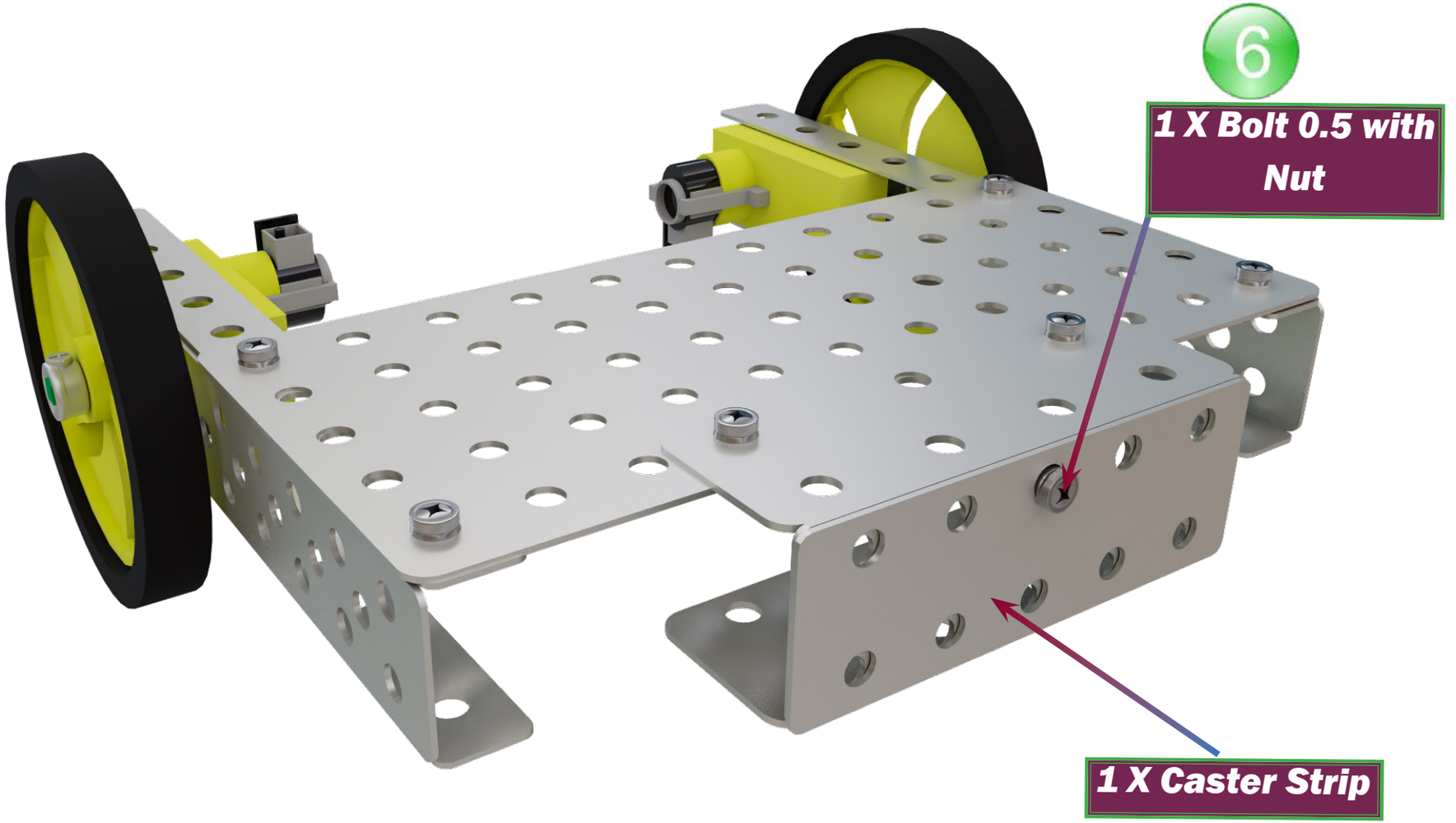


MECHANZ™

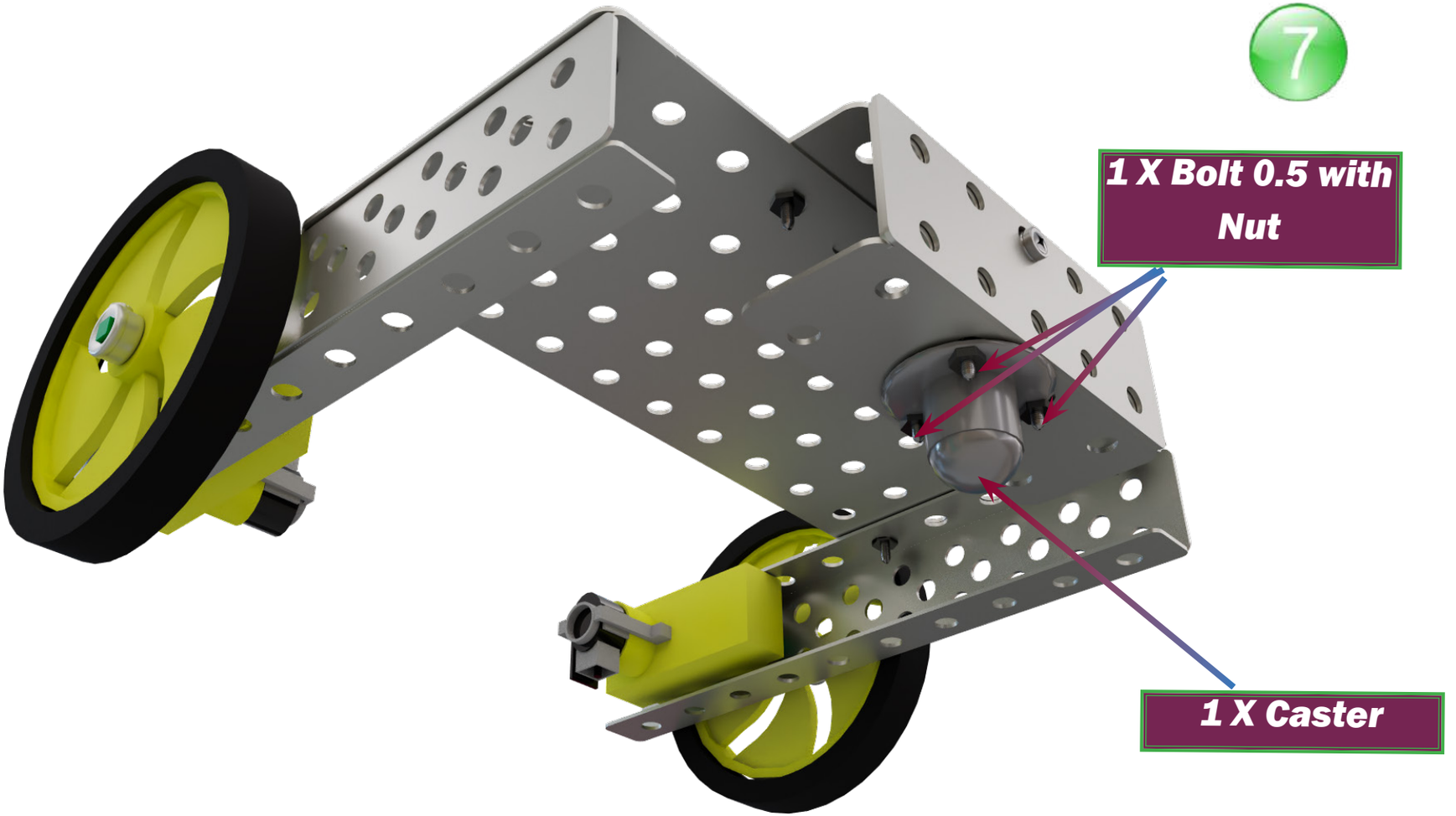


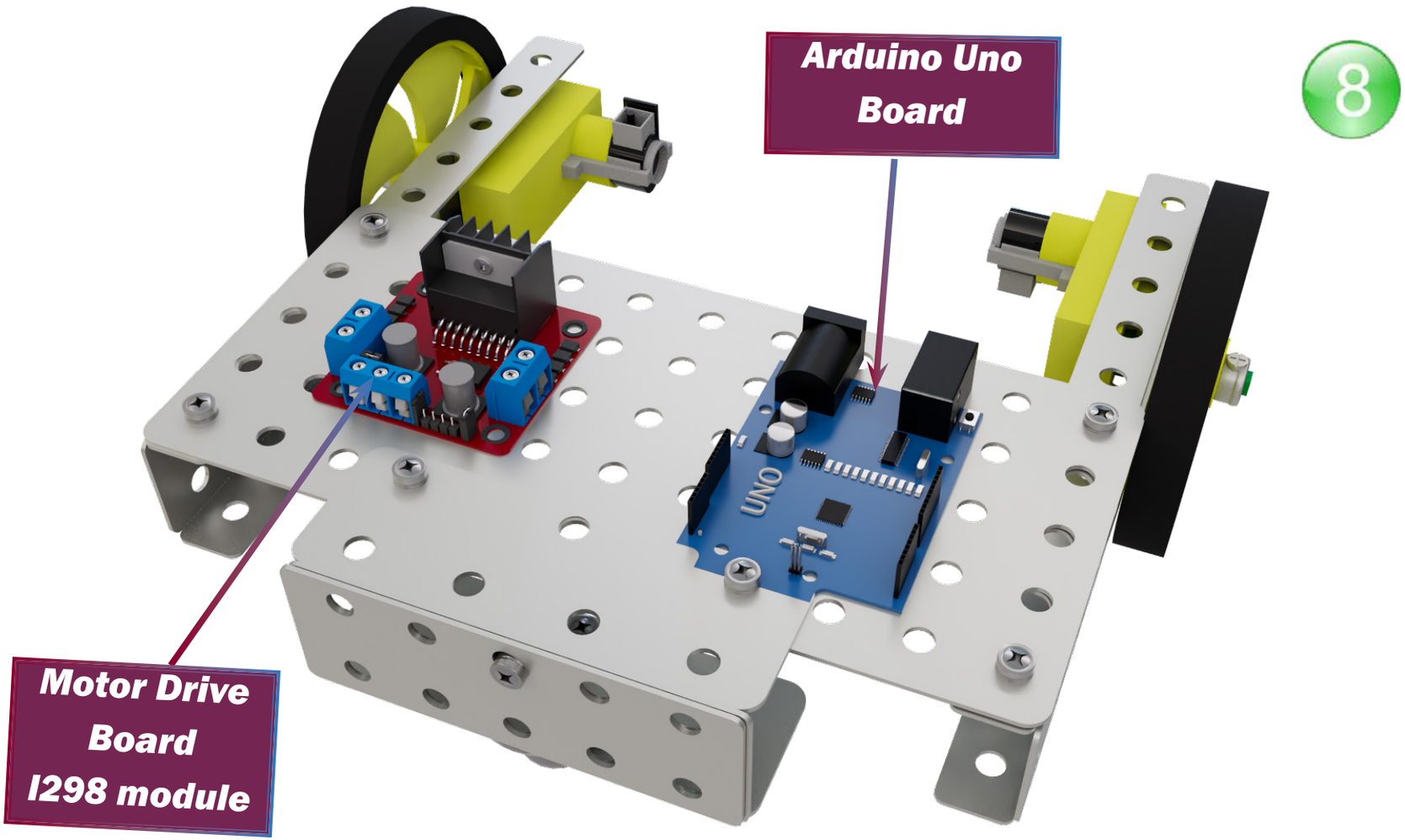
MECHANZ™





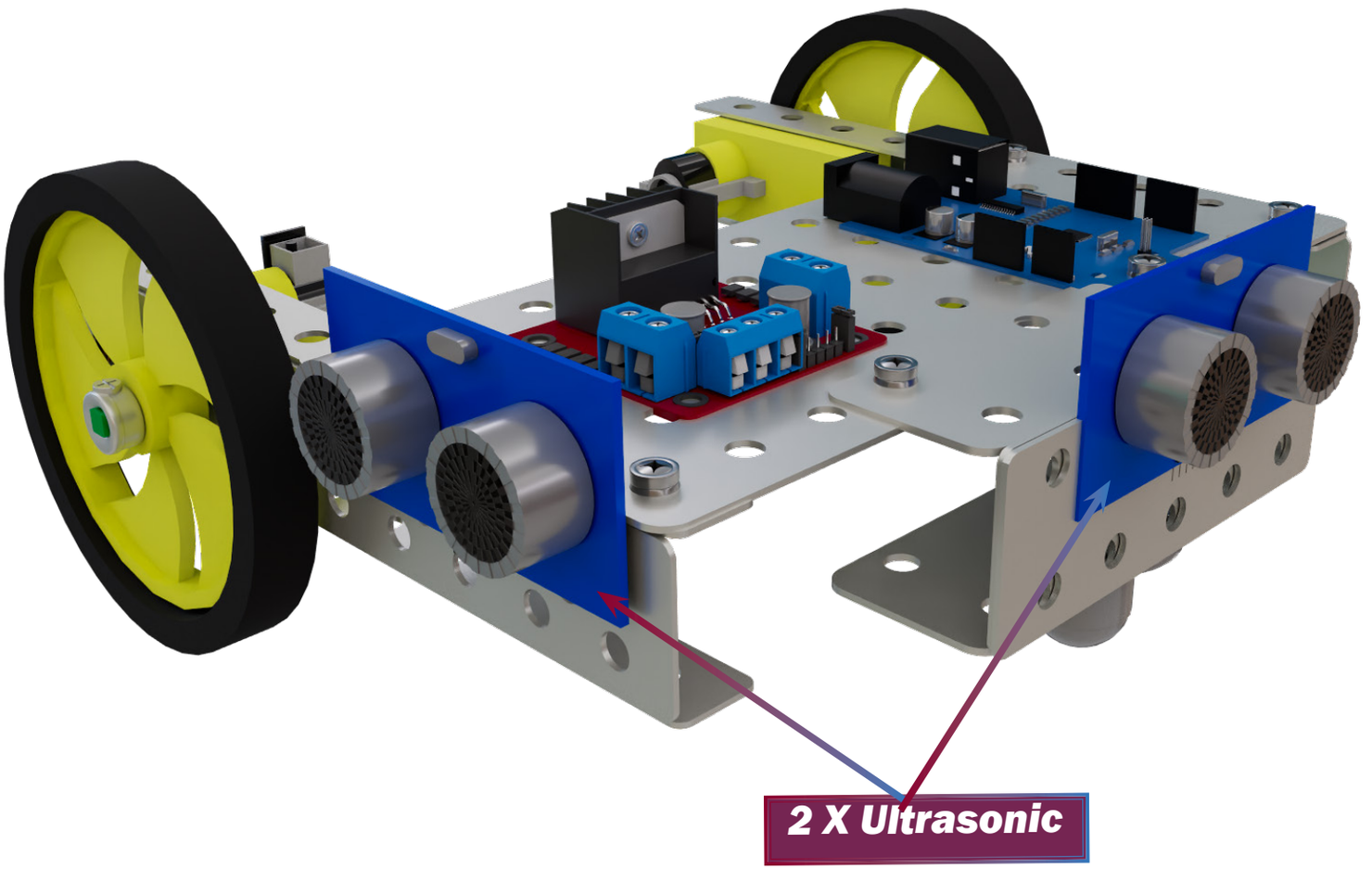
7



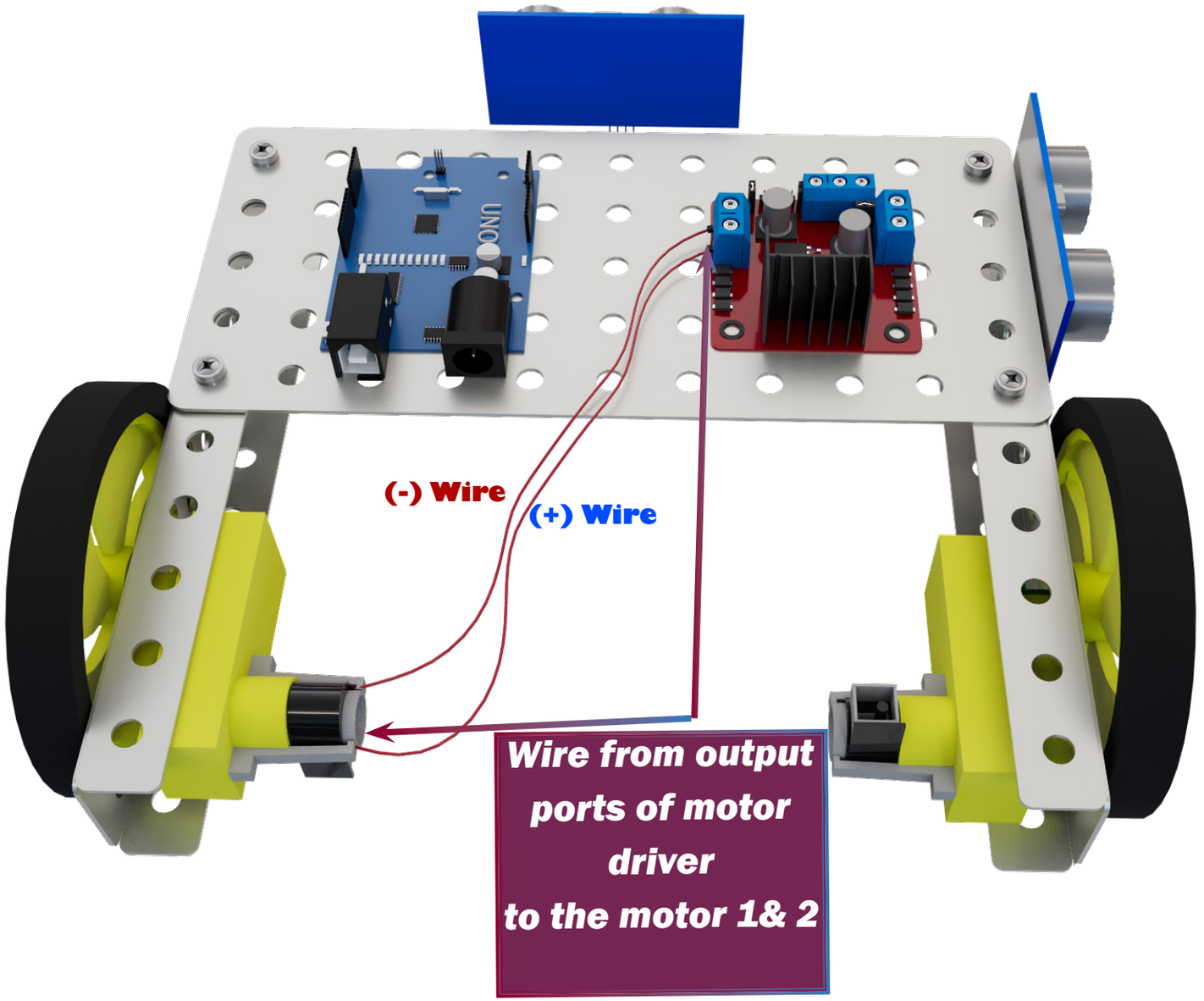


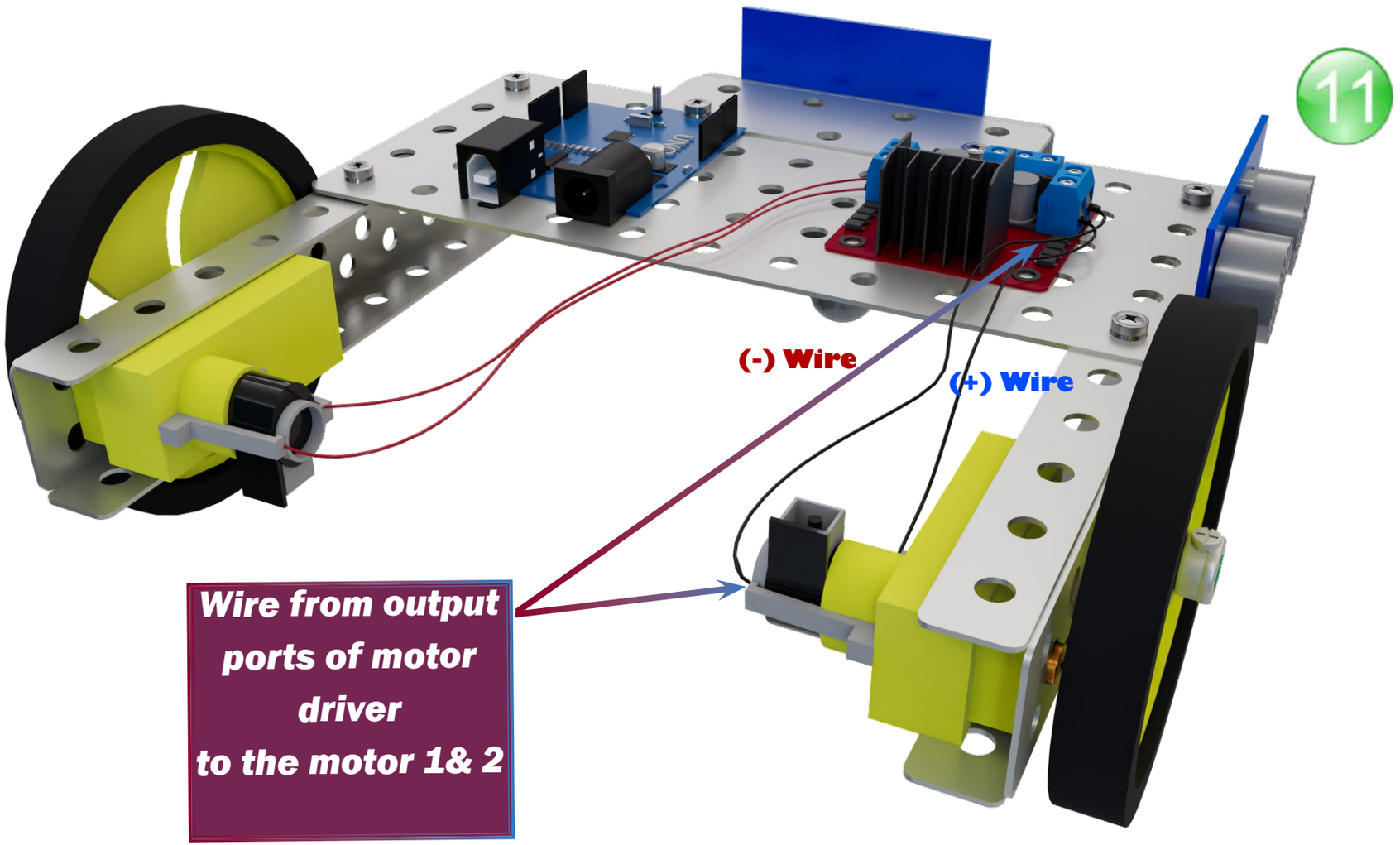
8

9

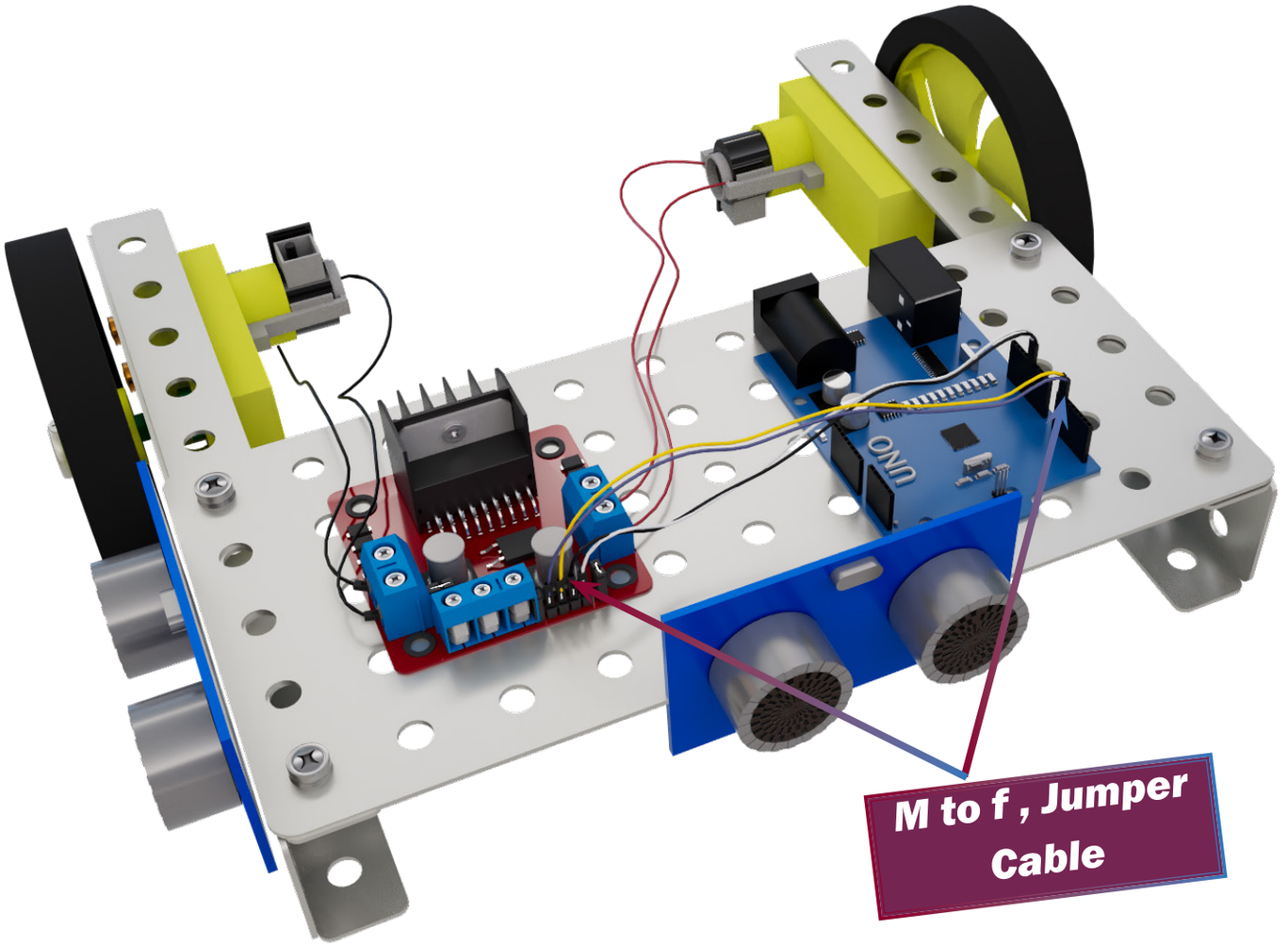


10

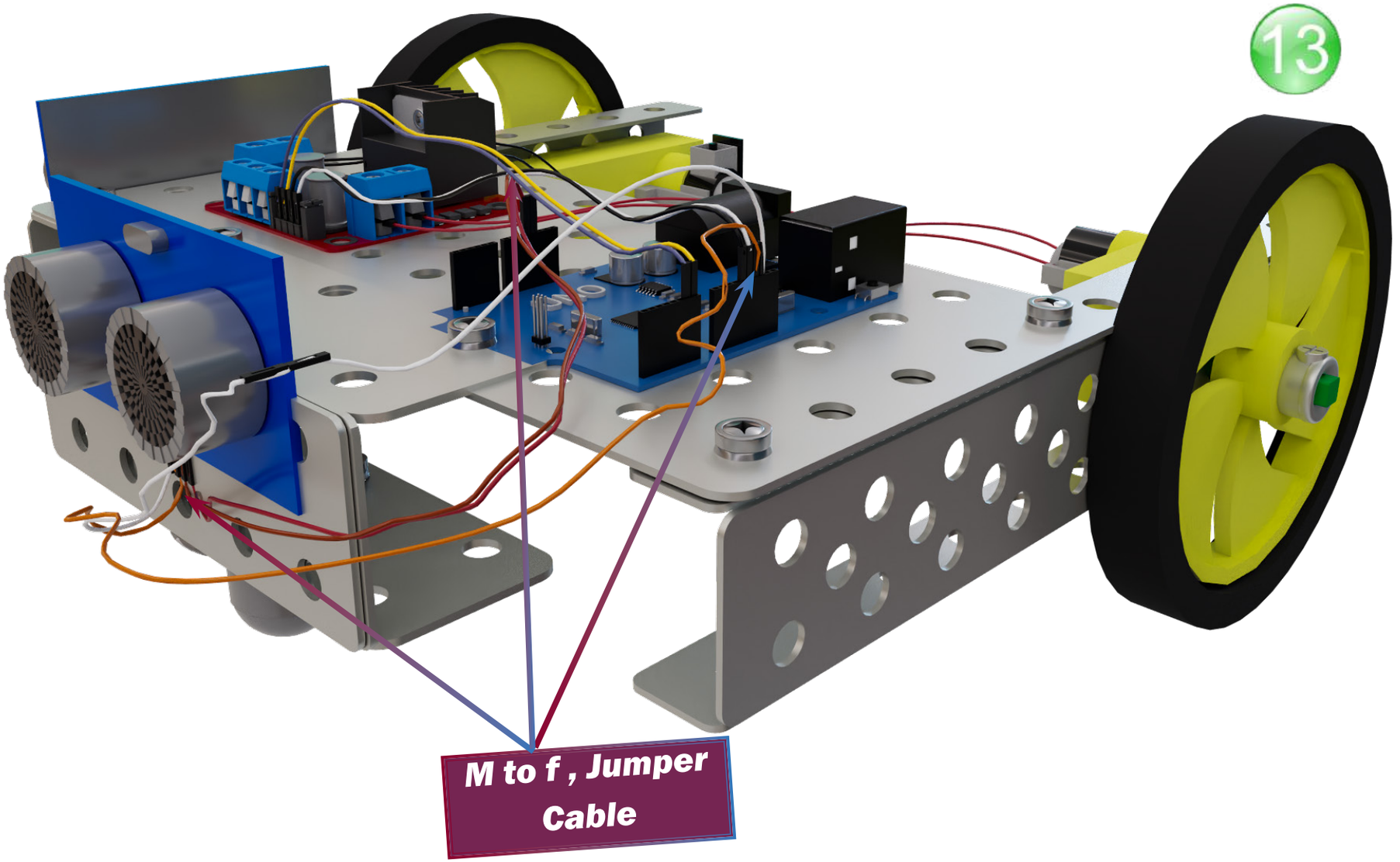


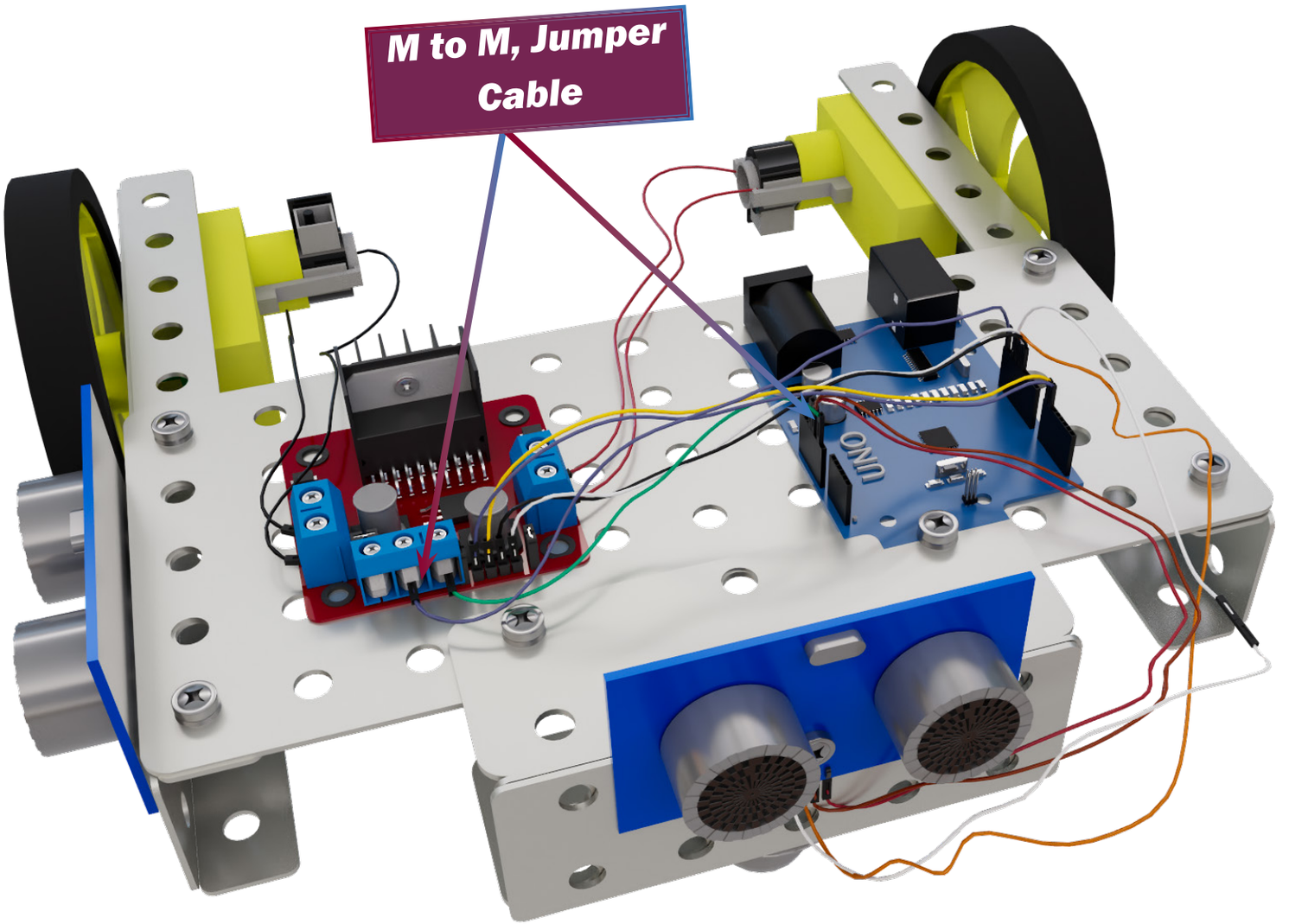


12

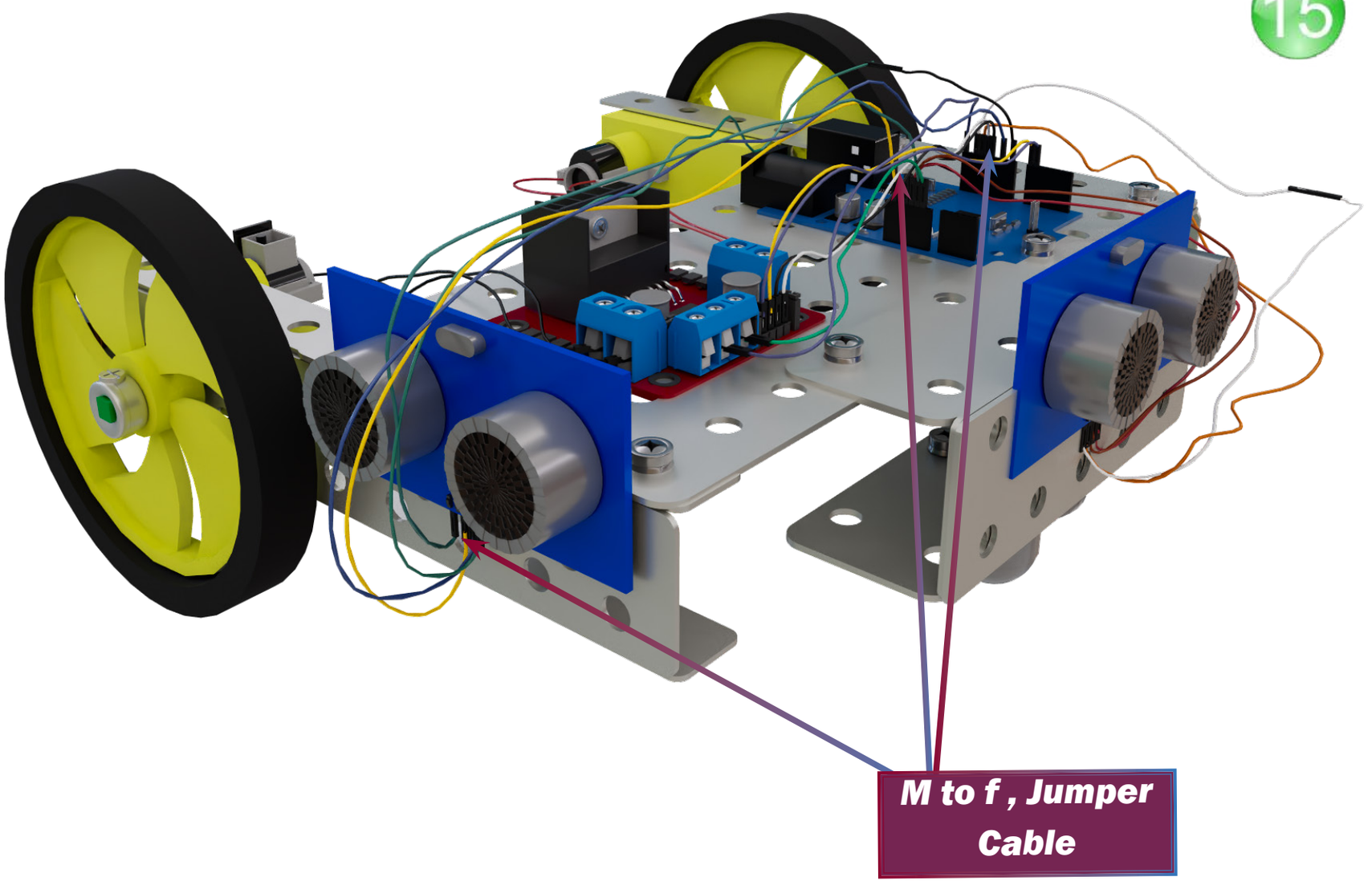


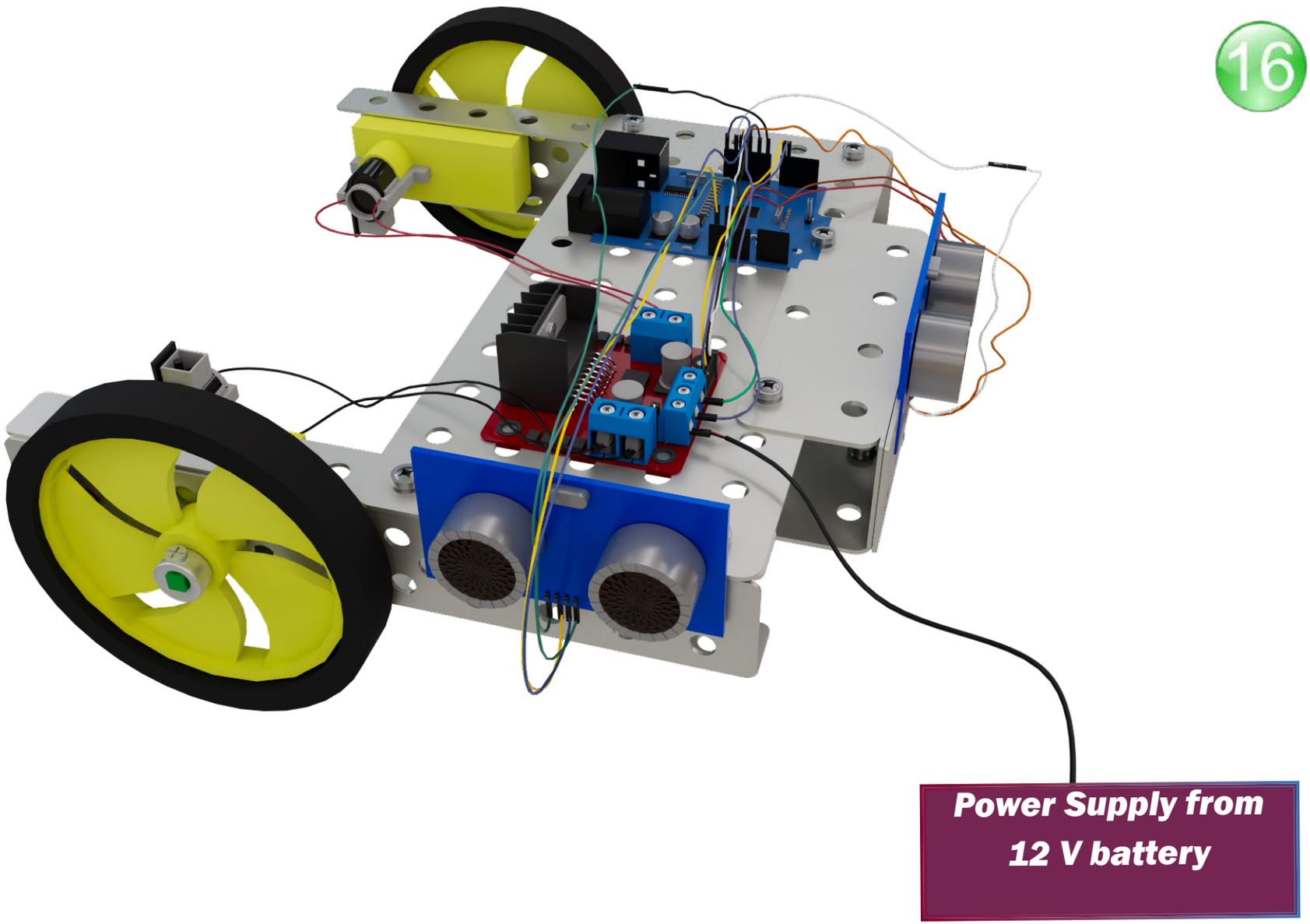
MECHANZ™





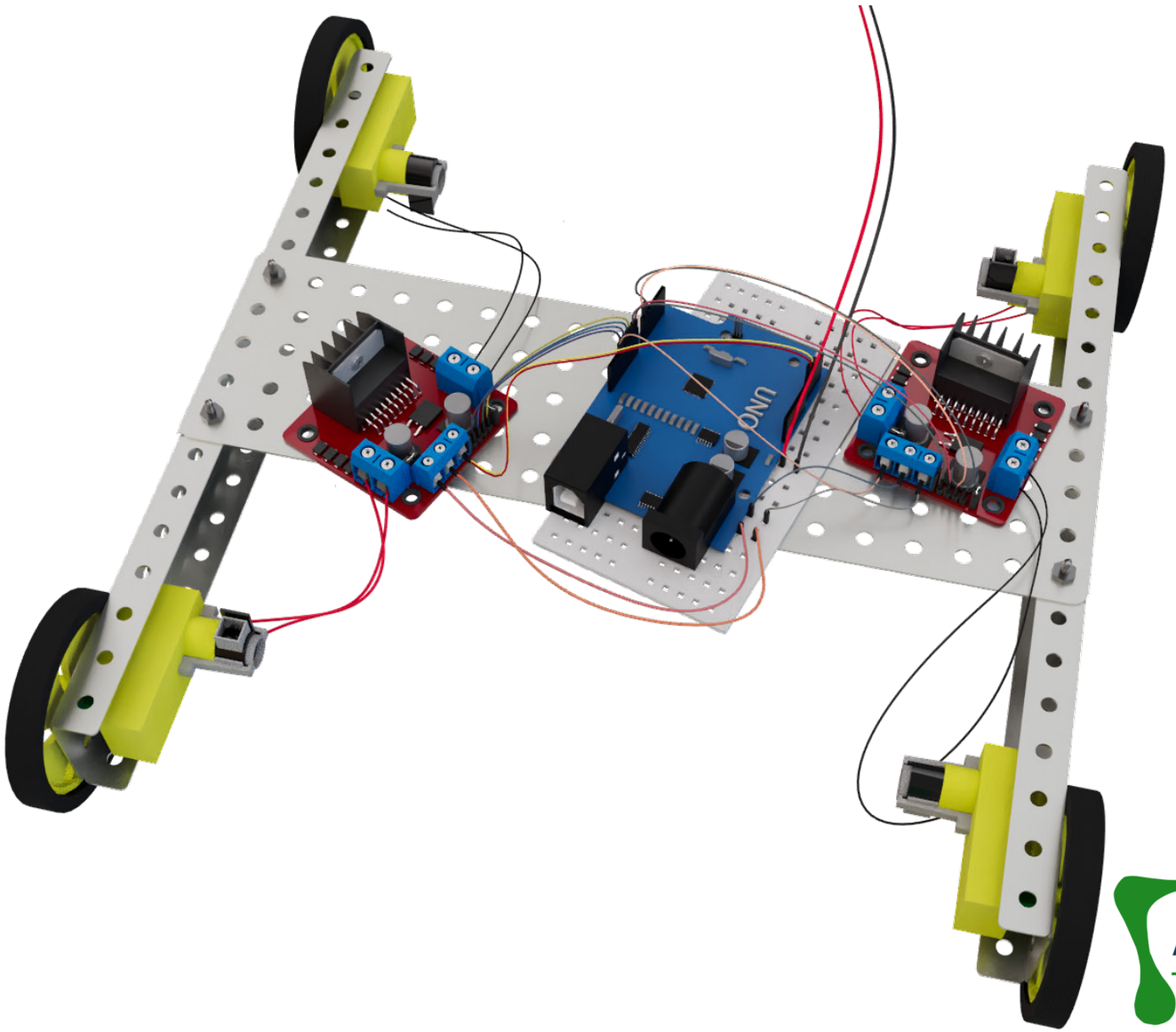
15





MECHANZ™

4 Wheel Drive

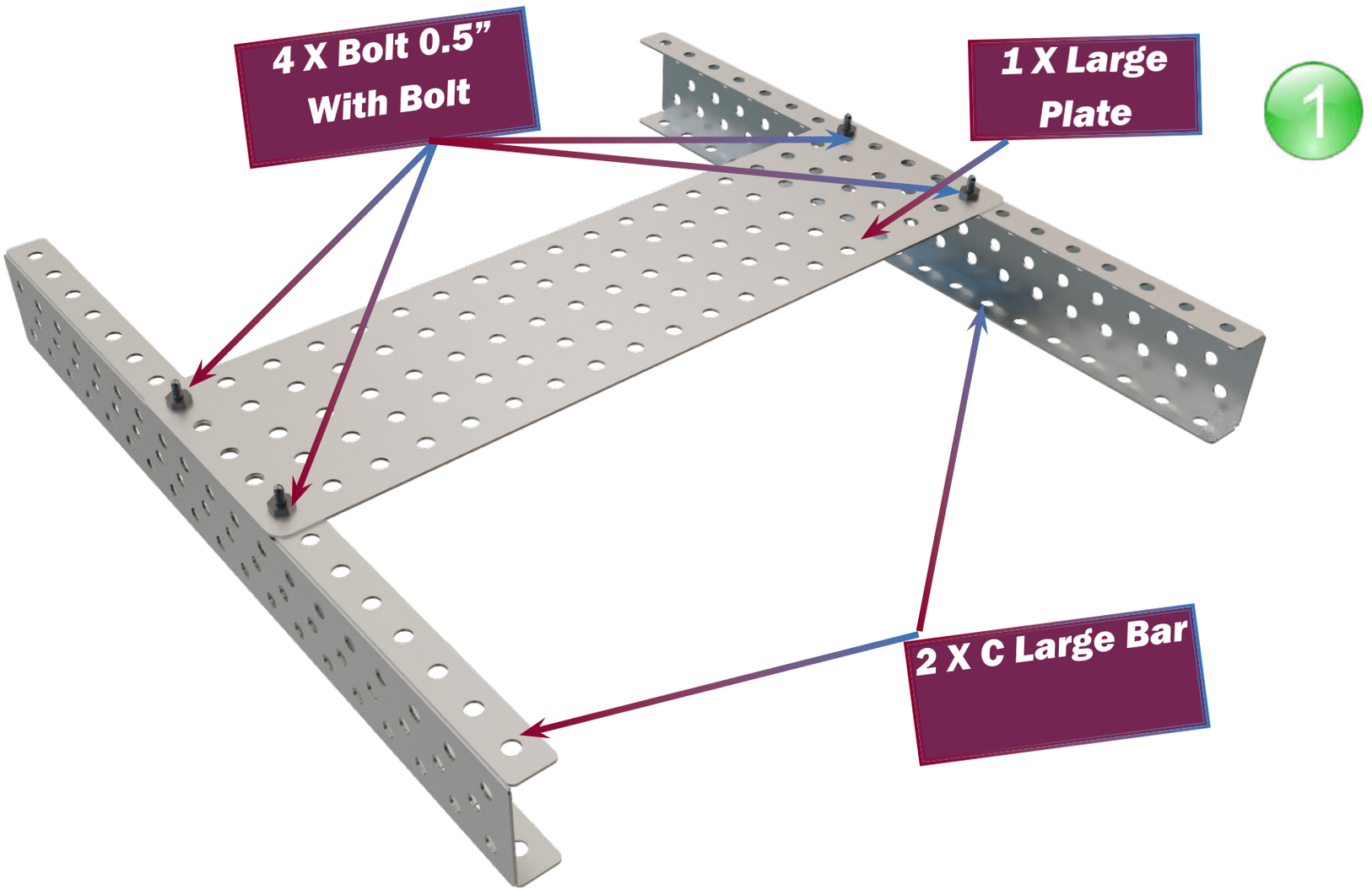


Product of

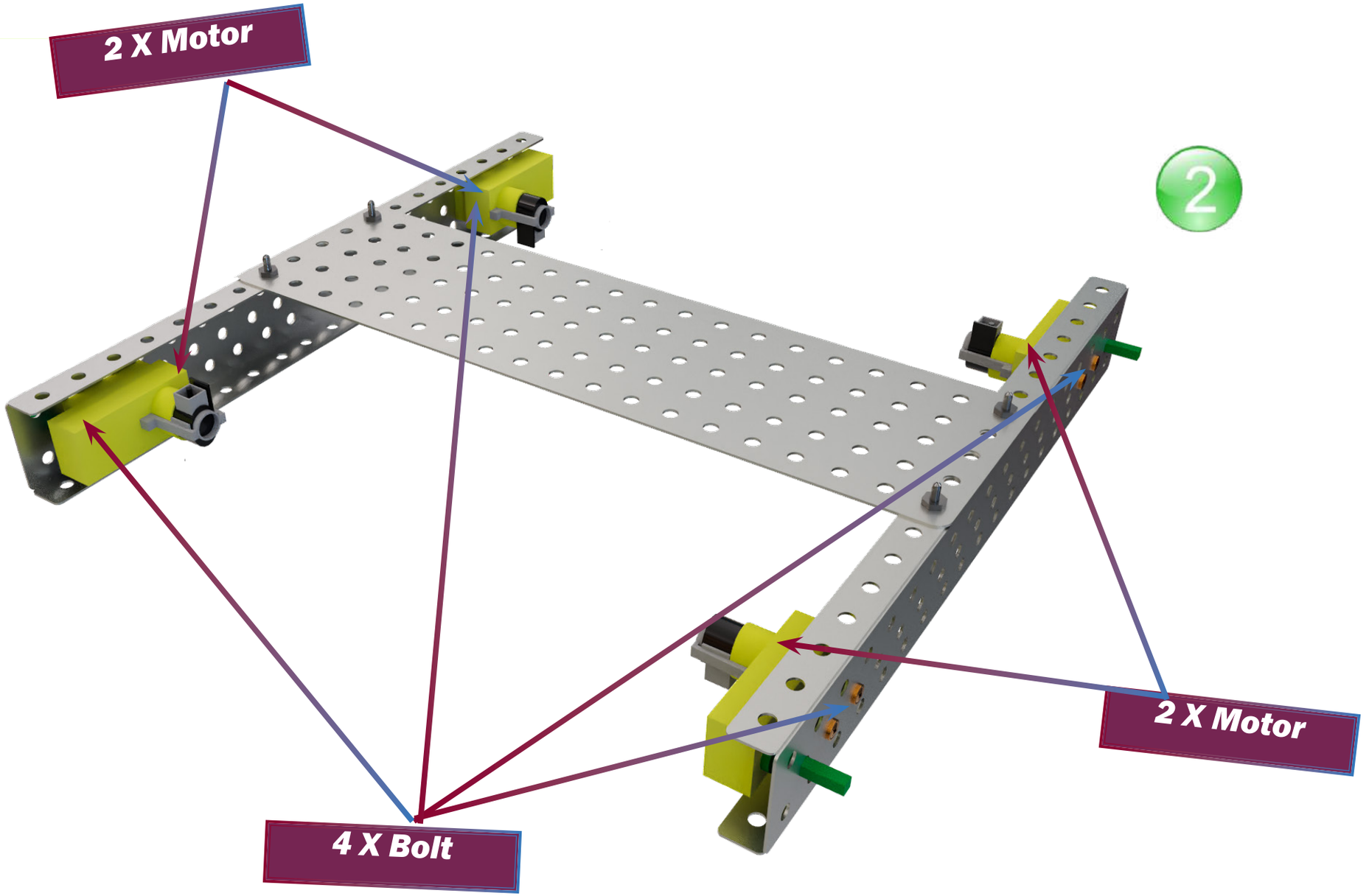
ABLEDUCATION™

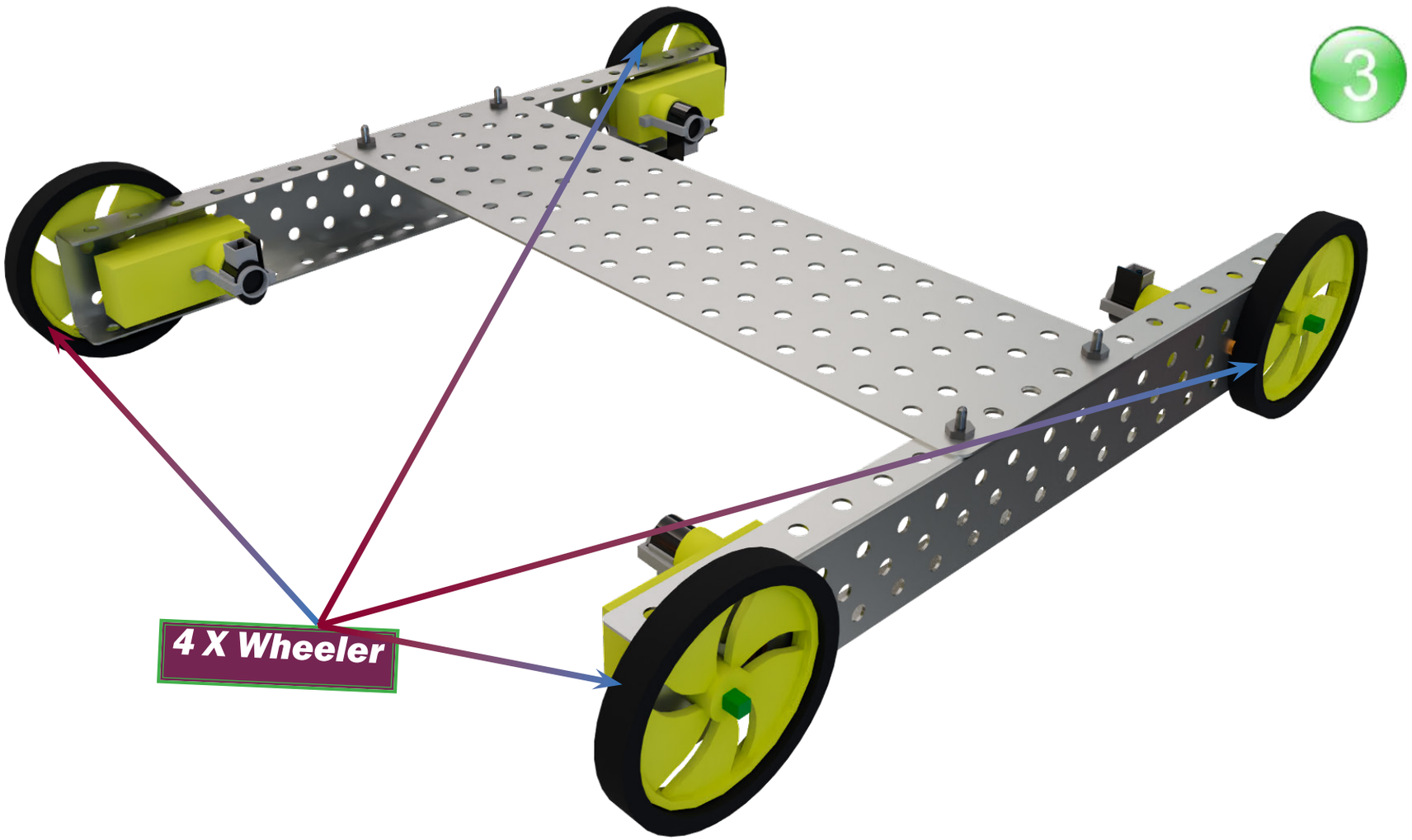
THINK • EXPLORE • CREATE • INNOVATE

MECHANZ™



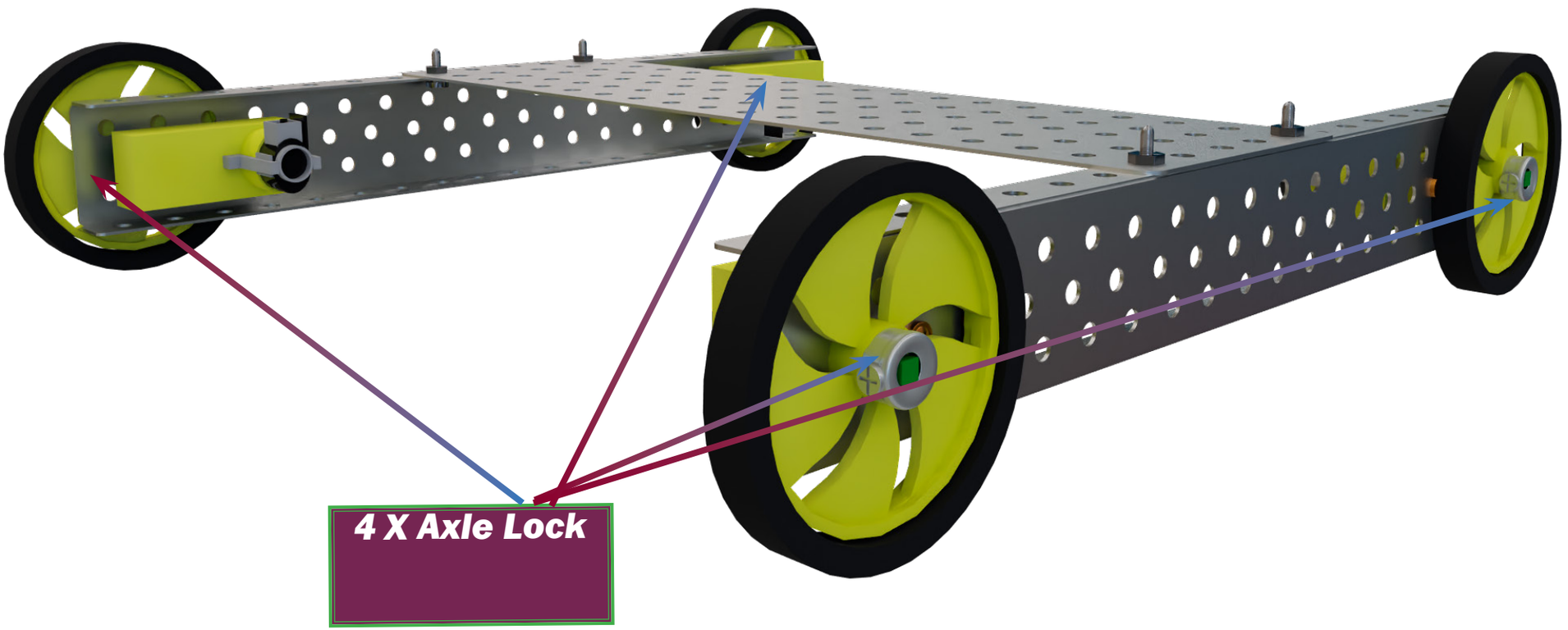
MECHANZ™



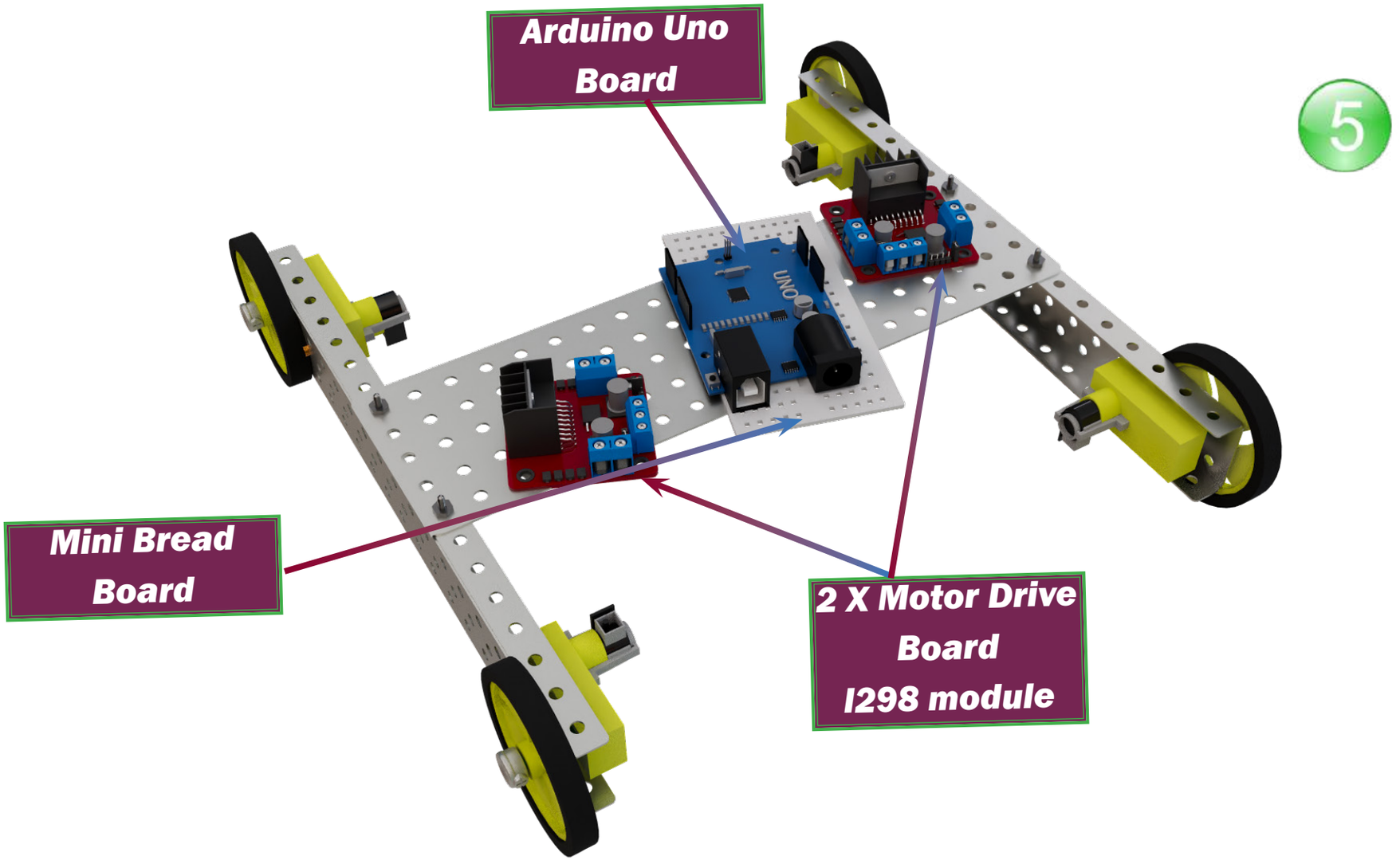


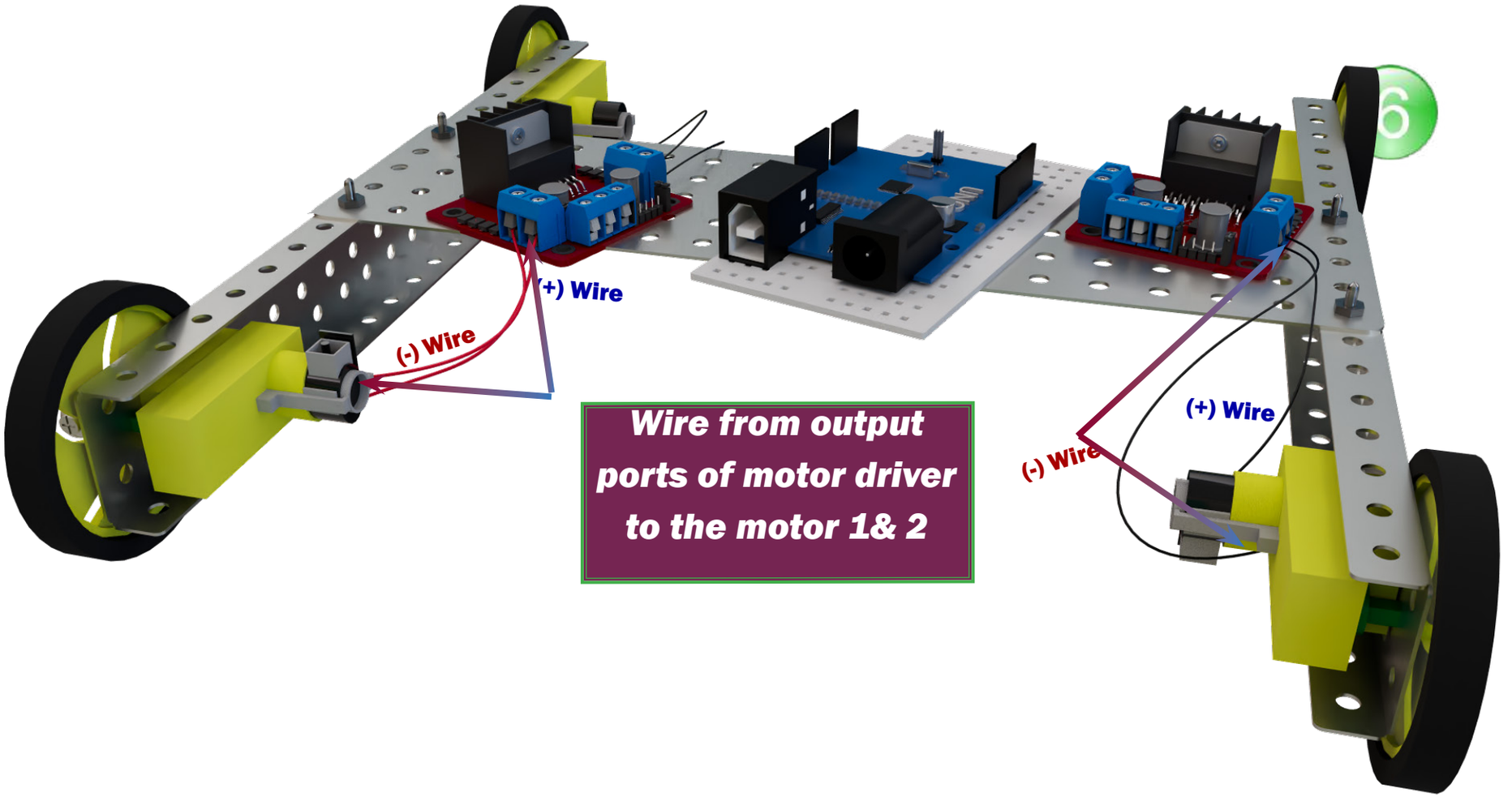
MECHANZ™

4



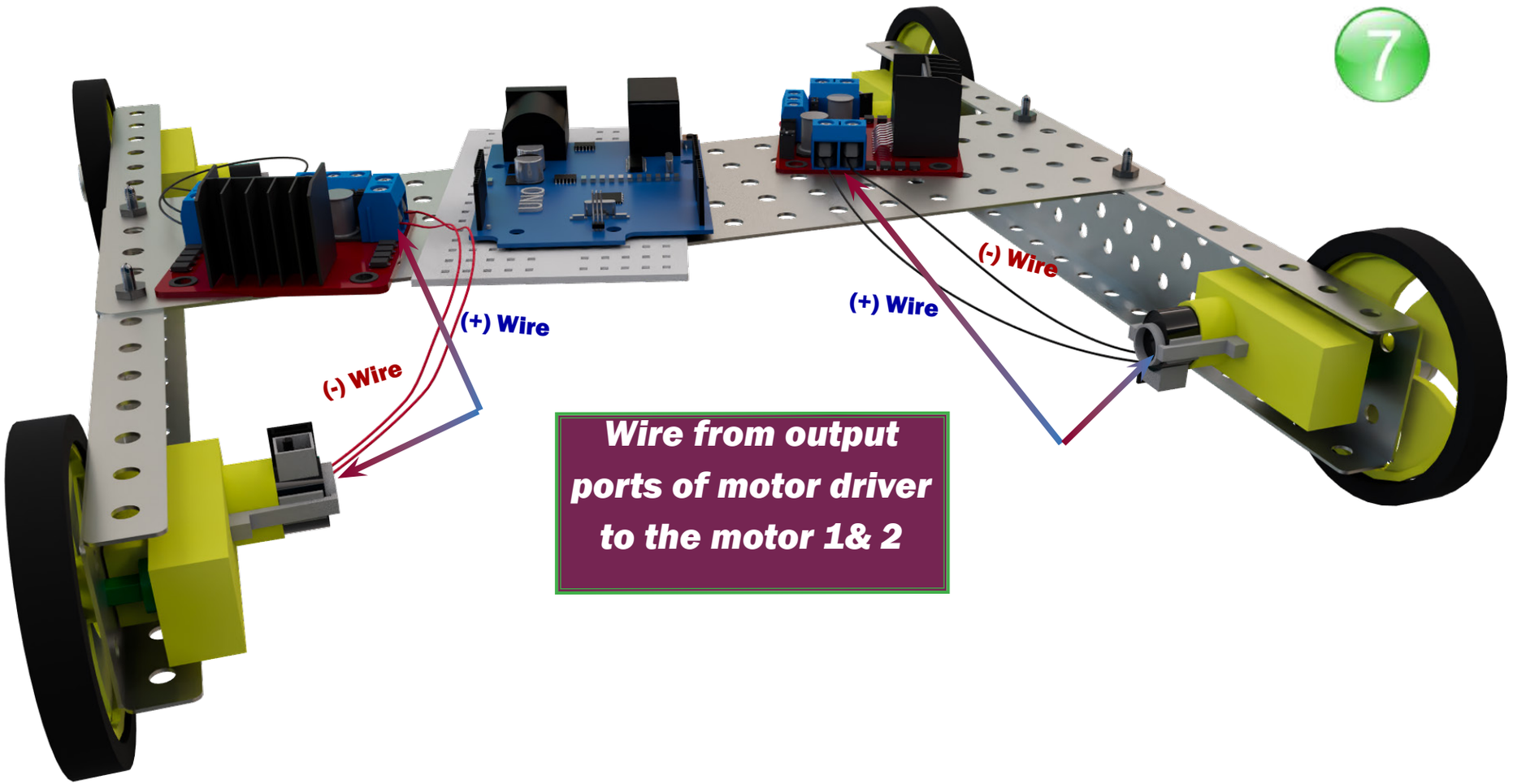
MECHANZ™


Arduino Uno Board
Mini Bread Board
2 X Motor Drive Board I298 module
5

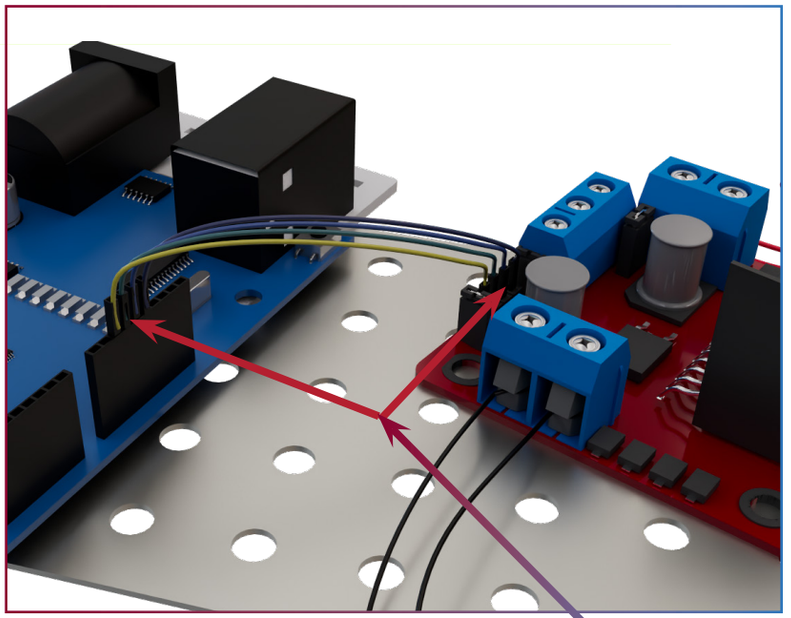


Wire from output ports of motor driver to the motor 1& 2

MECHANZ™



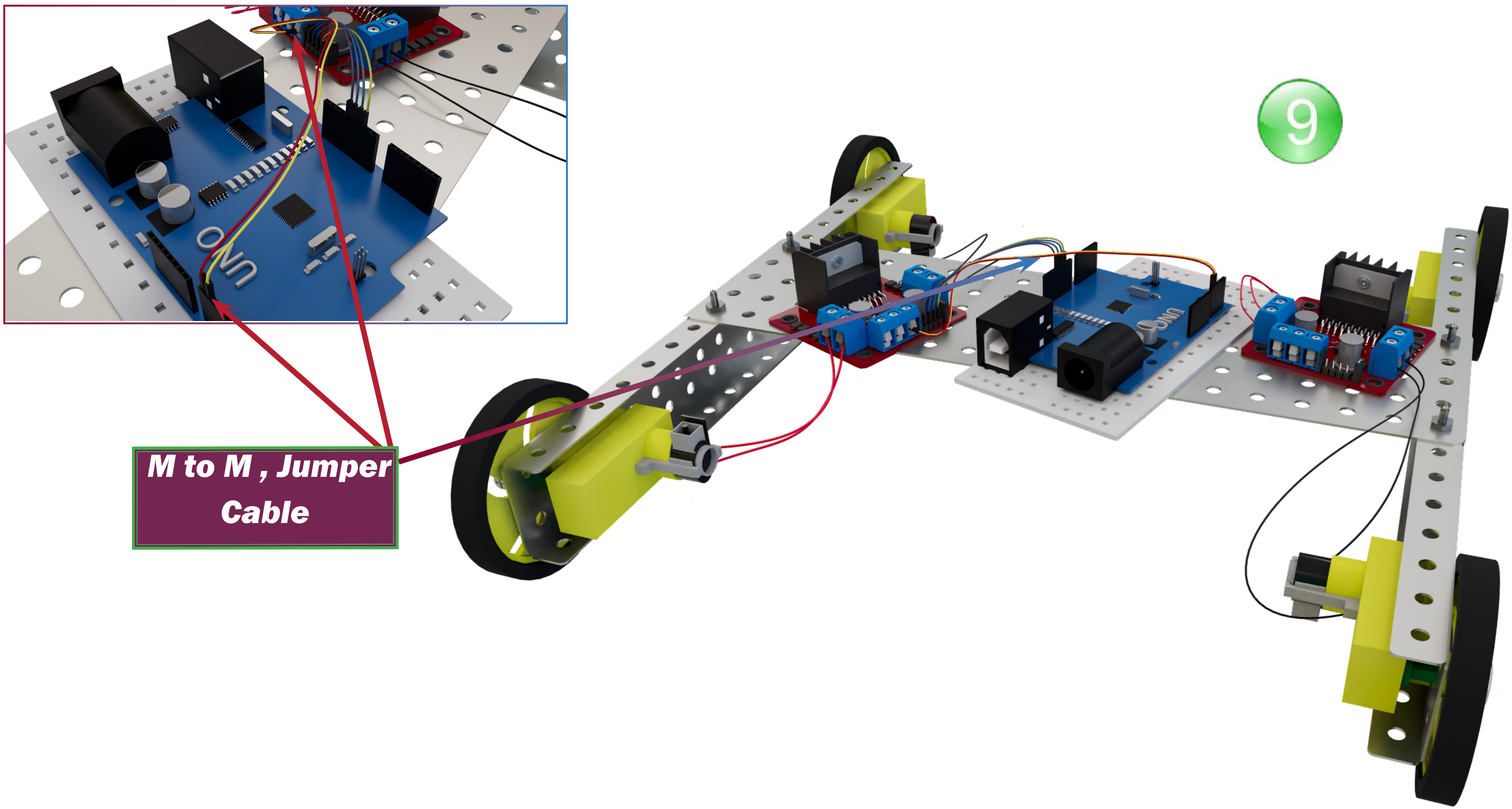
MECHANZ™

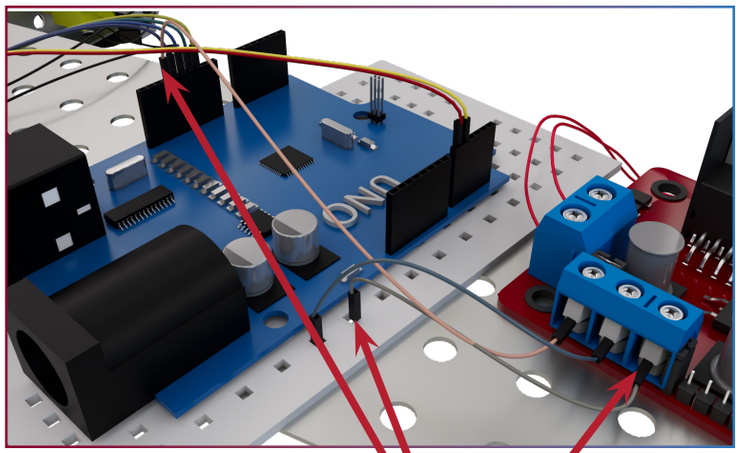


8

M to M , Jumper Cable

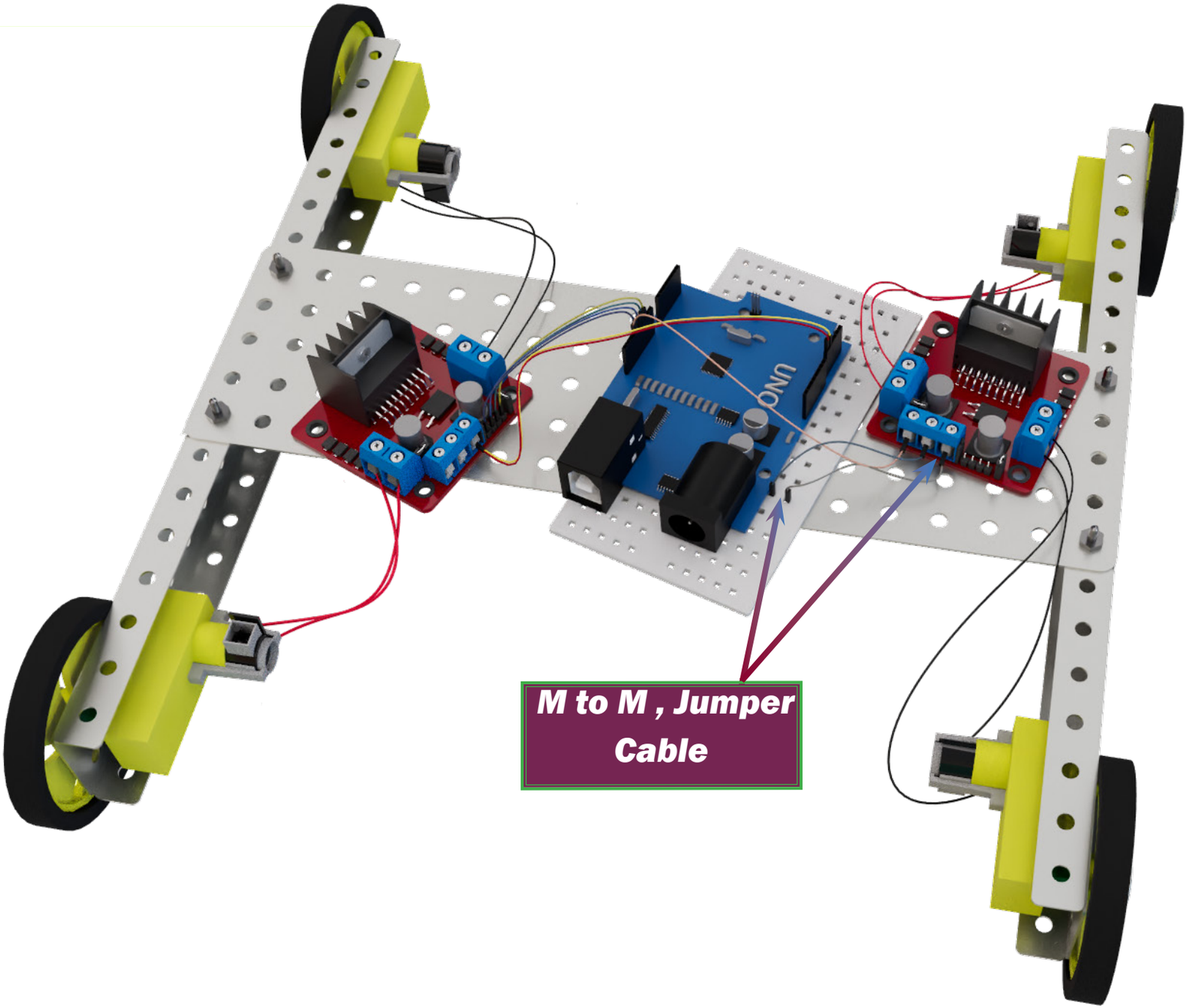
MECHANZ™



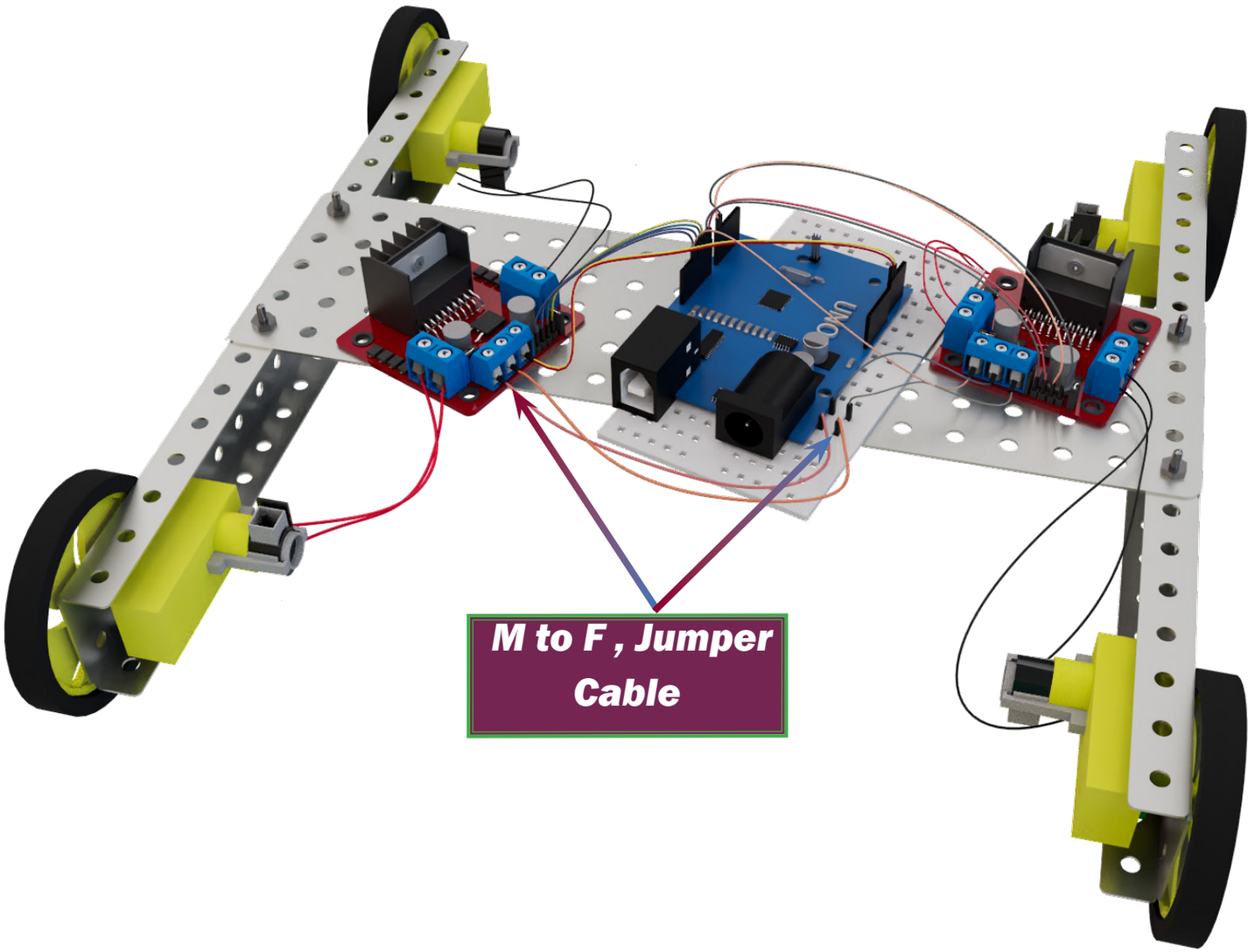


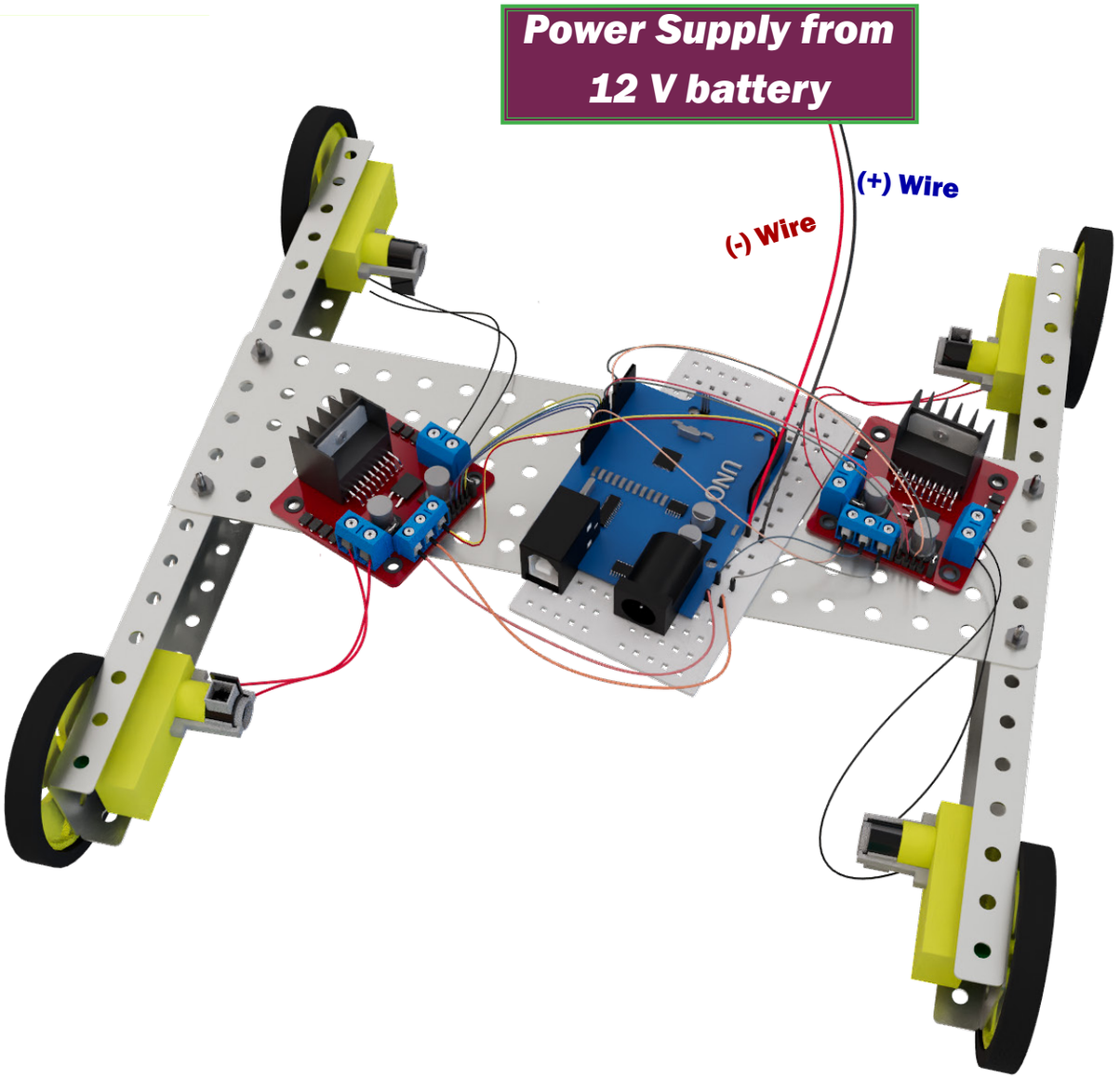
M to M , Jumper Cable

10



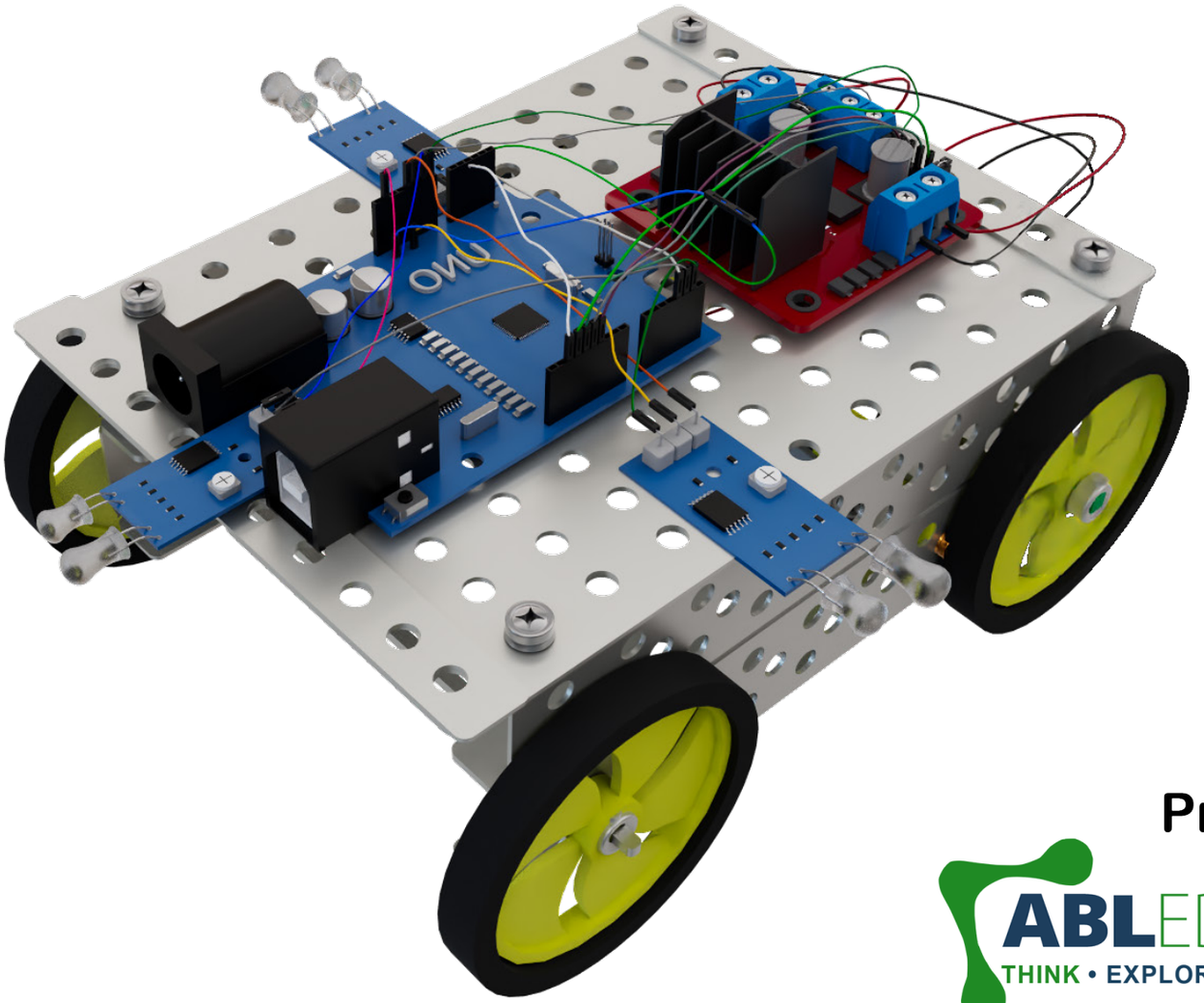
11





MECHANZ™

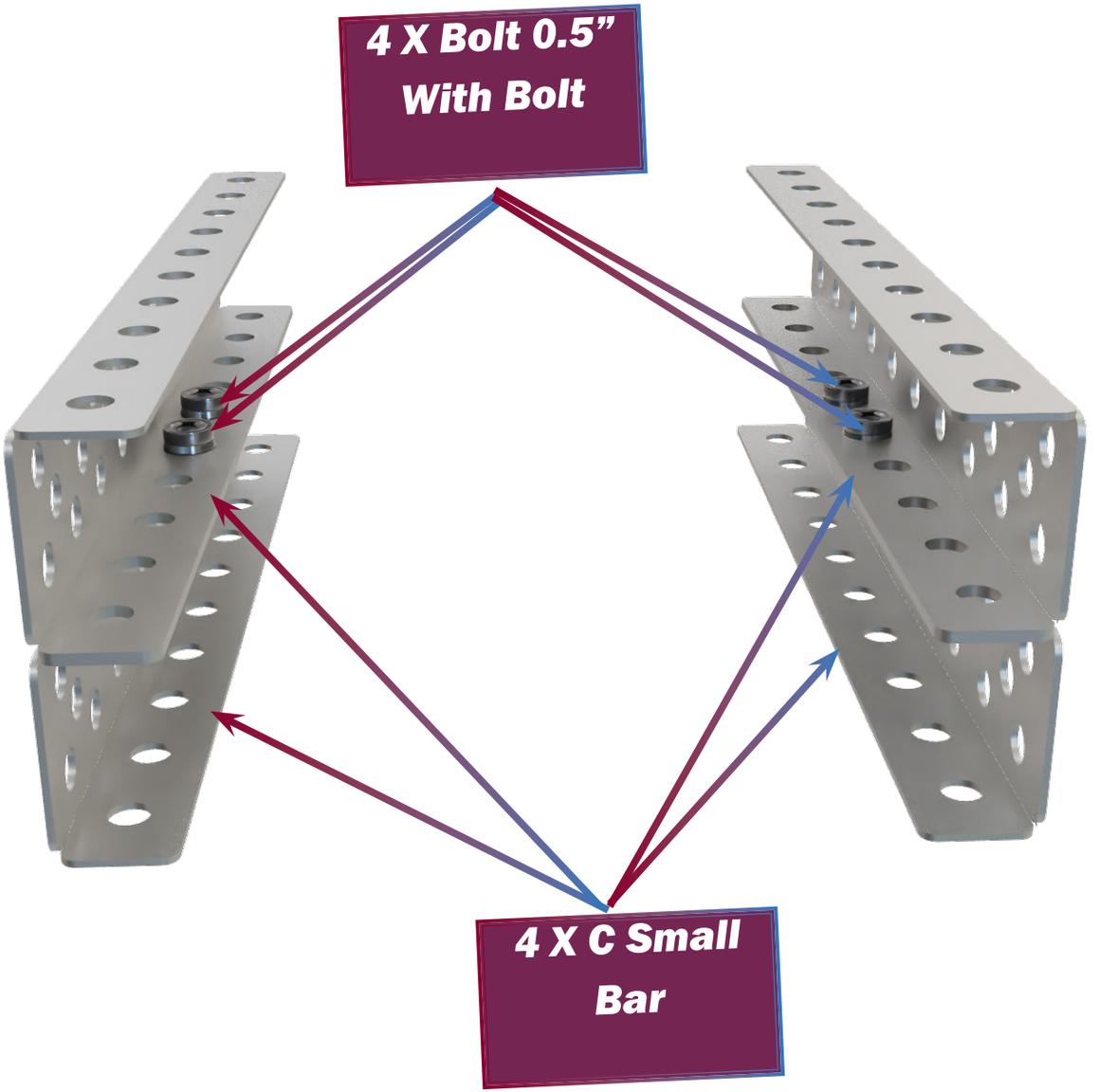
Light Following Bot



Product of

ABLEDUCATION™

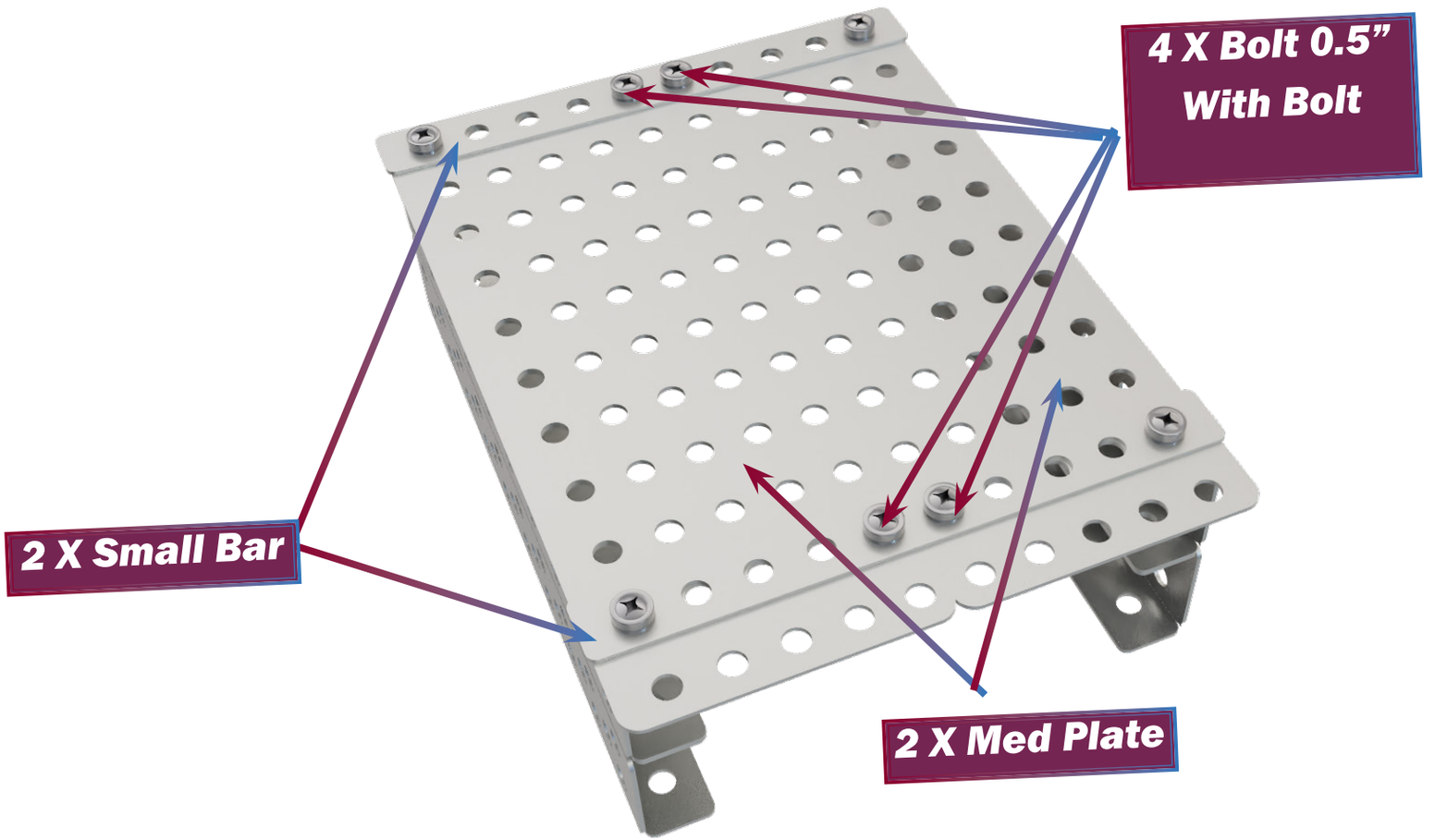
THINK • EXPLORE • CREATE • INNOVATE



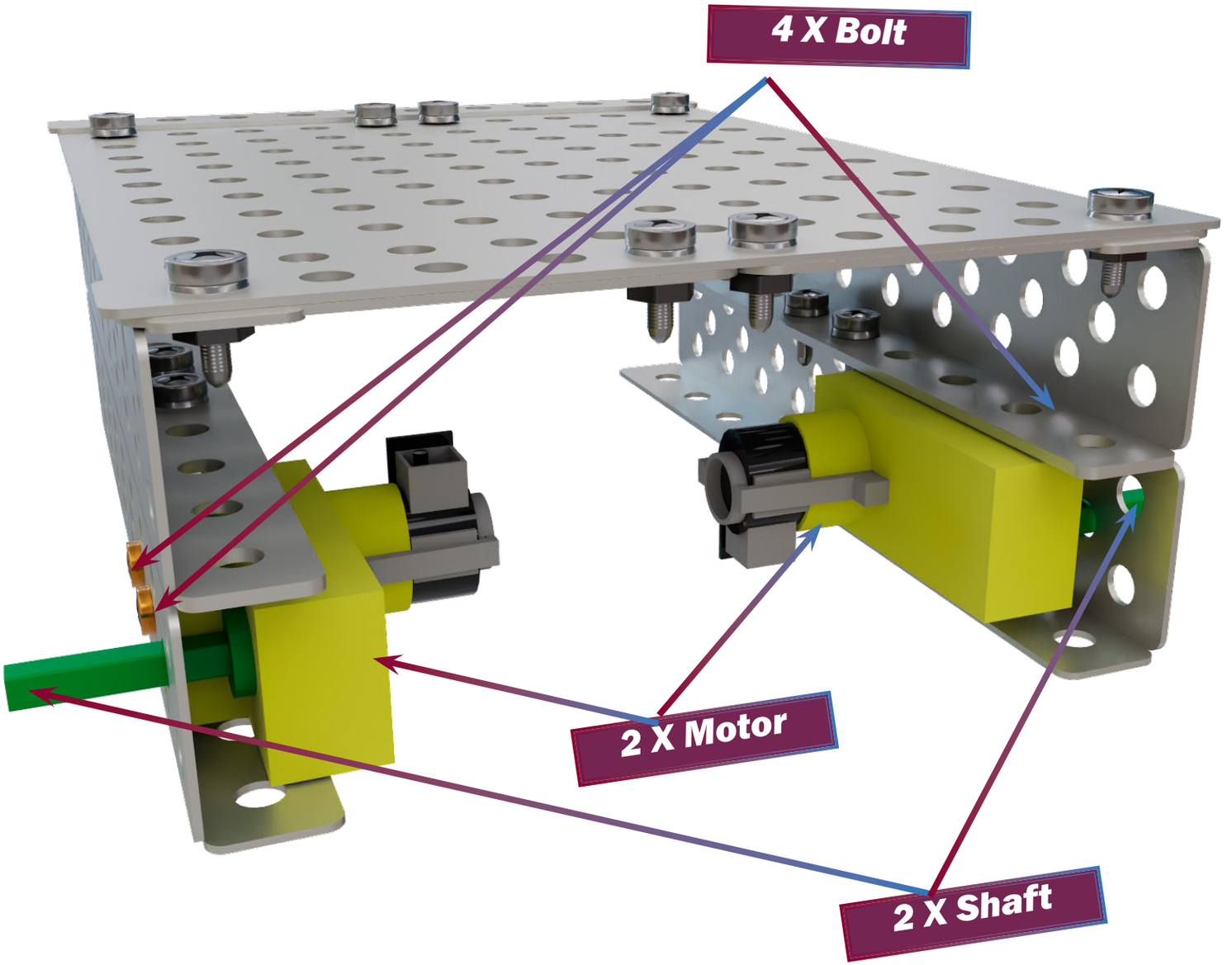
1

MECHANZ™

2

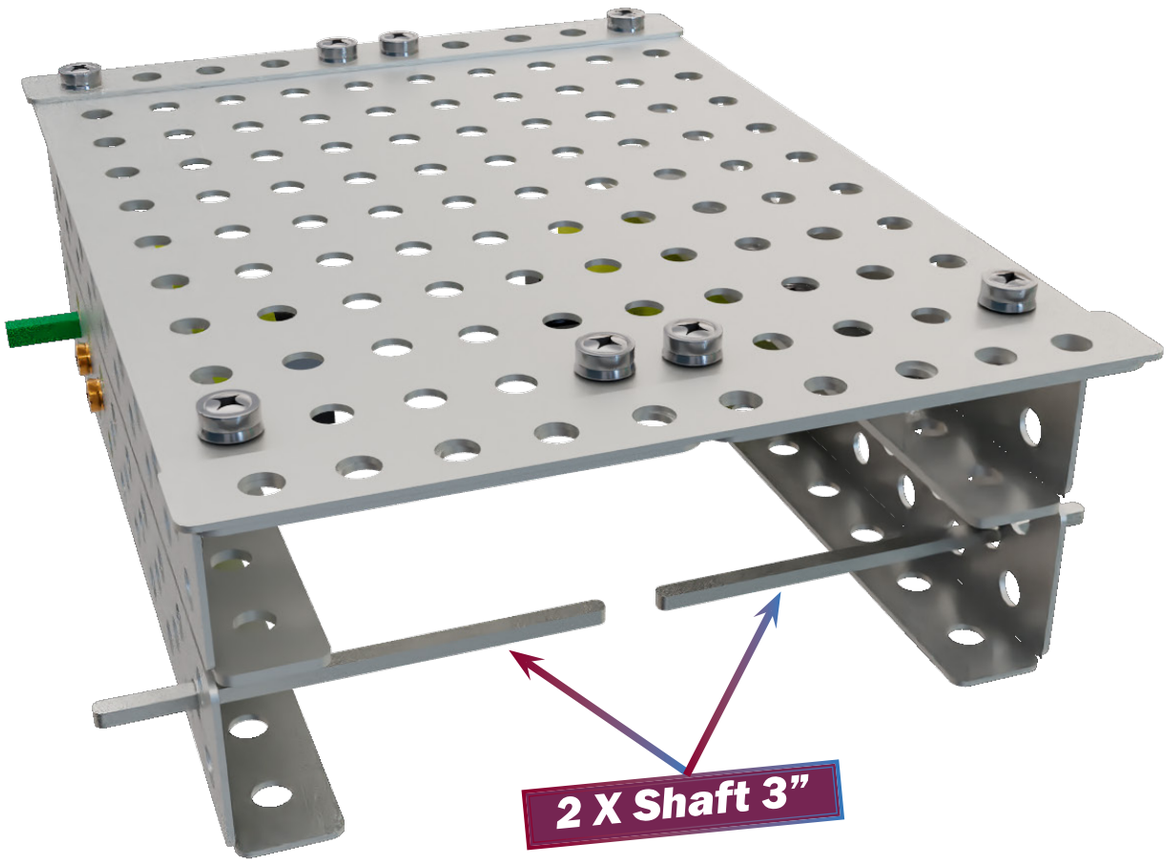


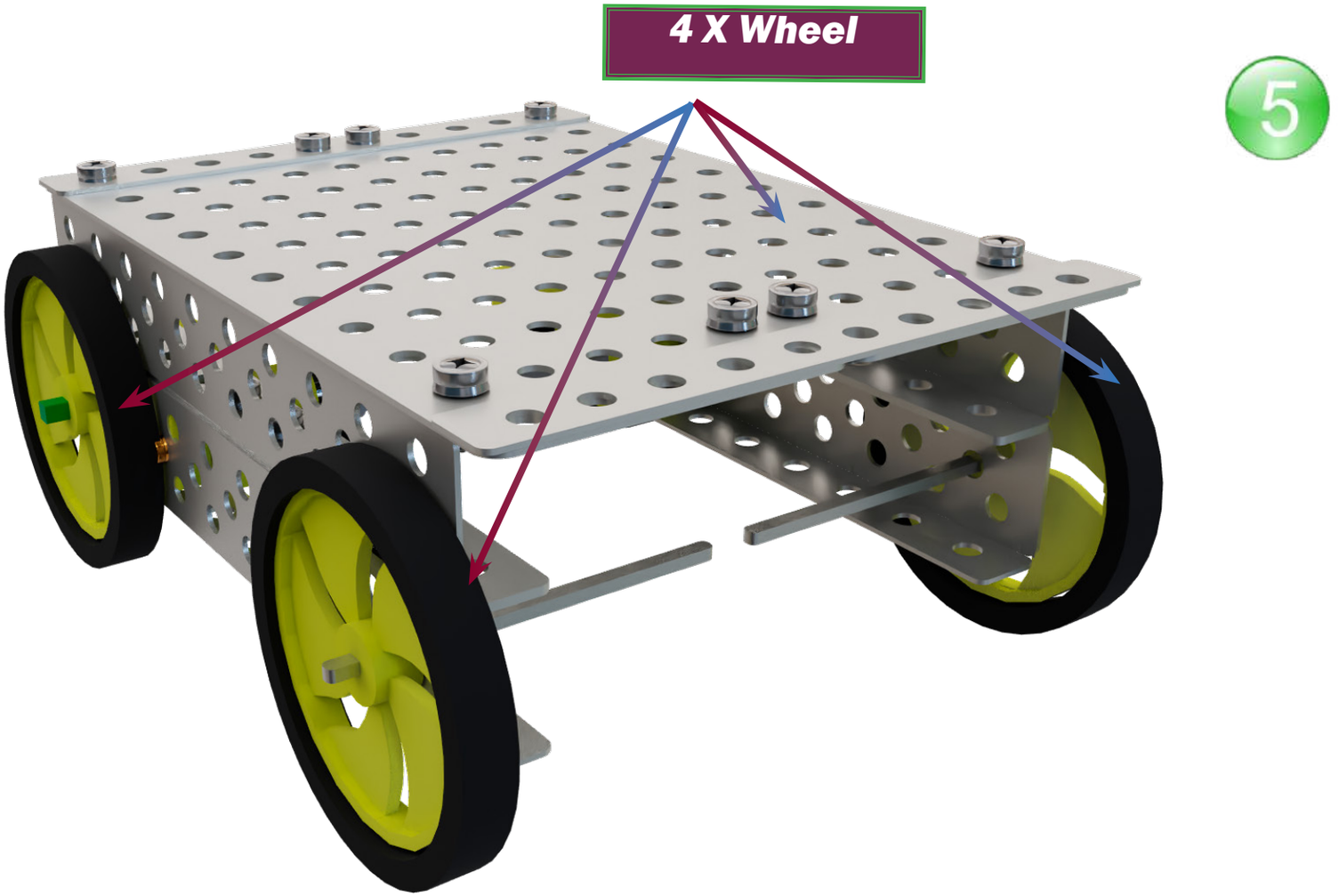
MECHANZ™

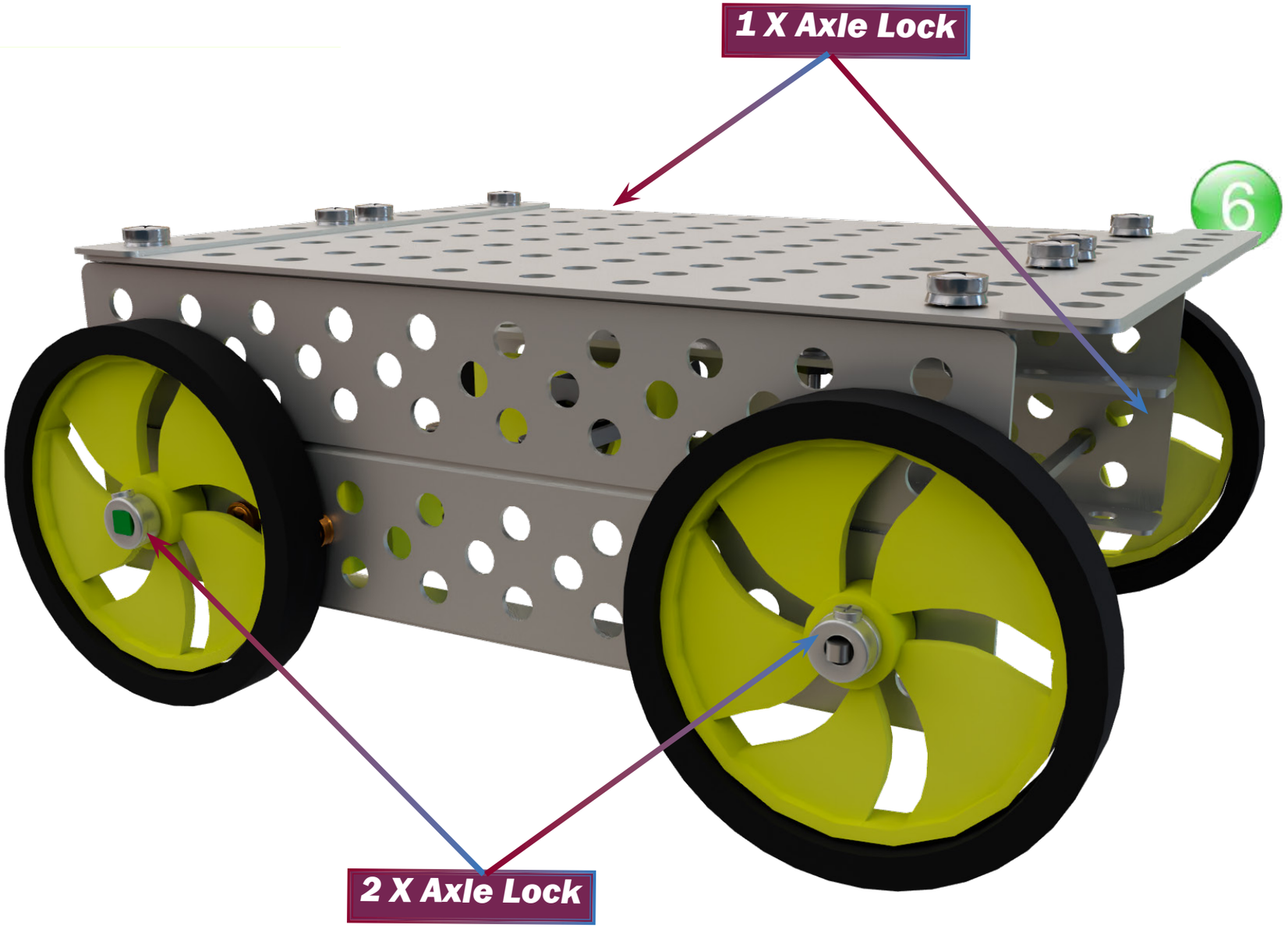


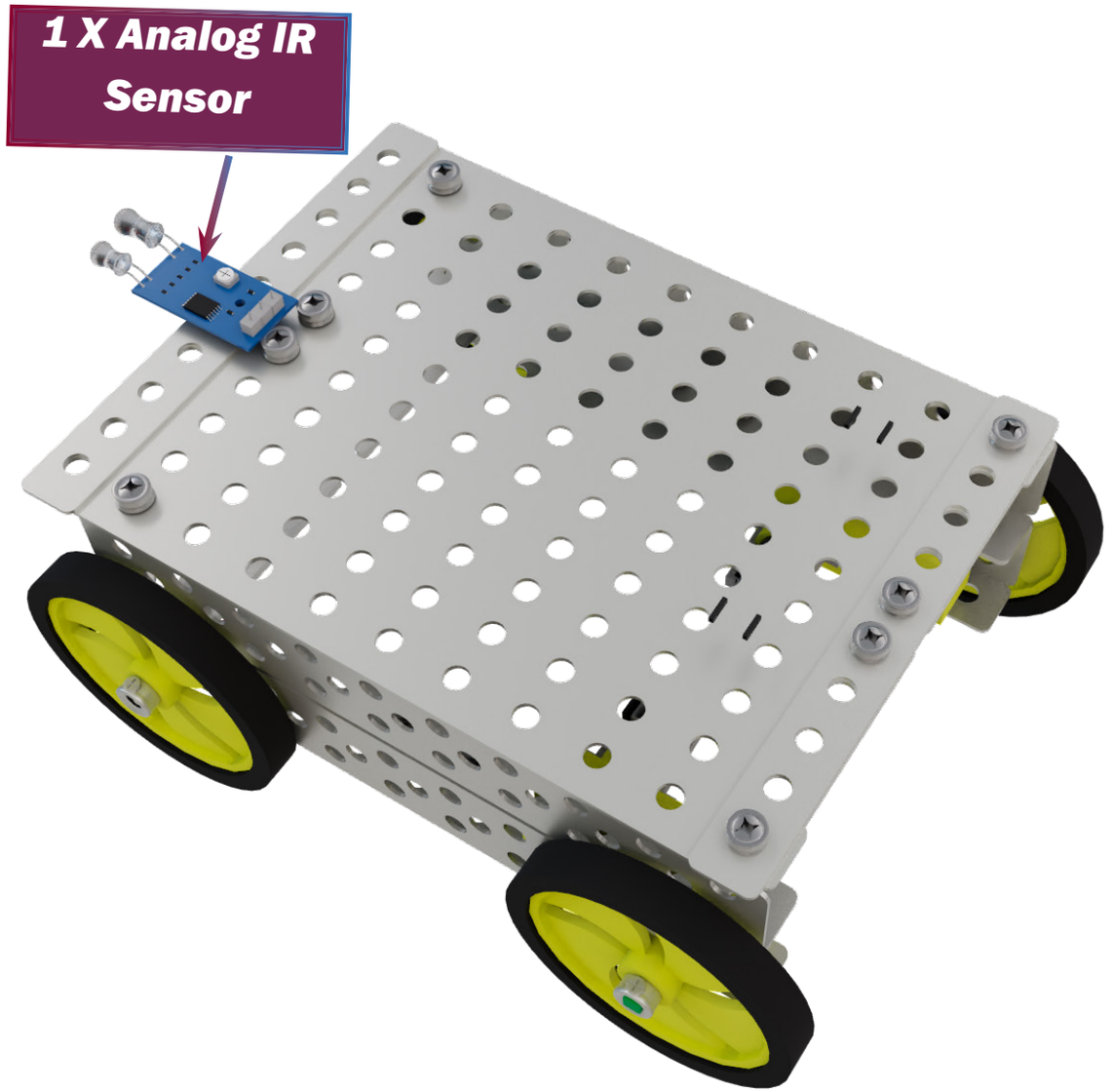
MECHANZ™

4

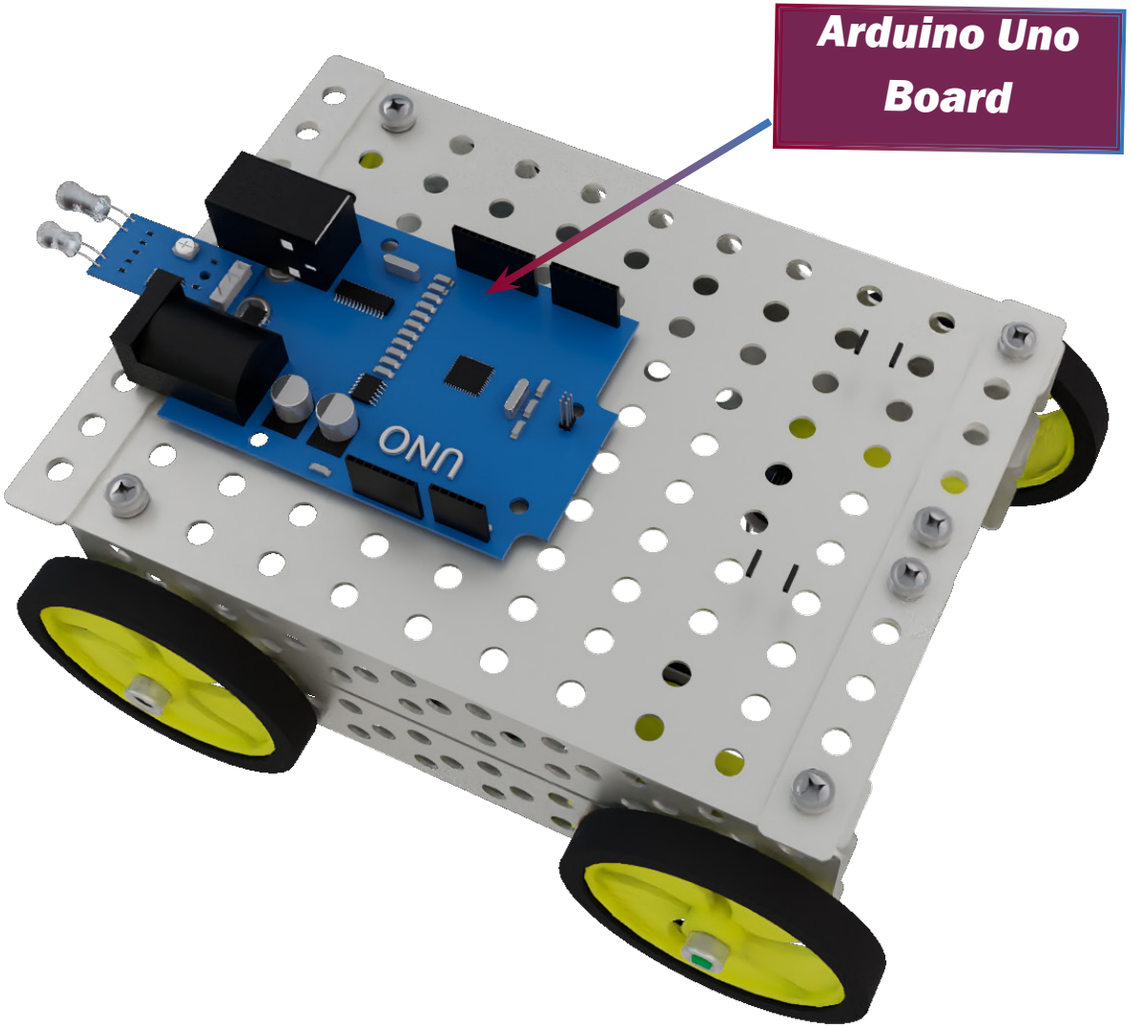


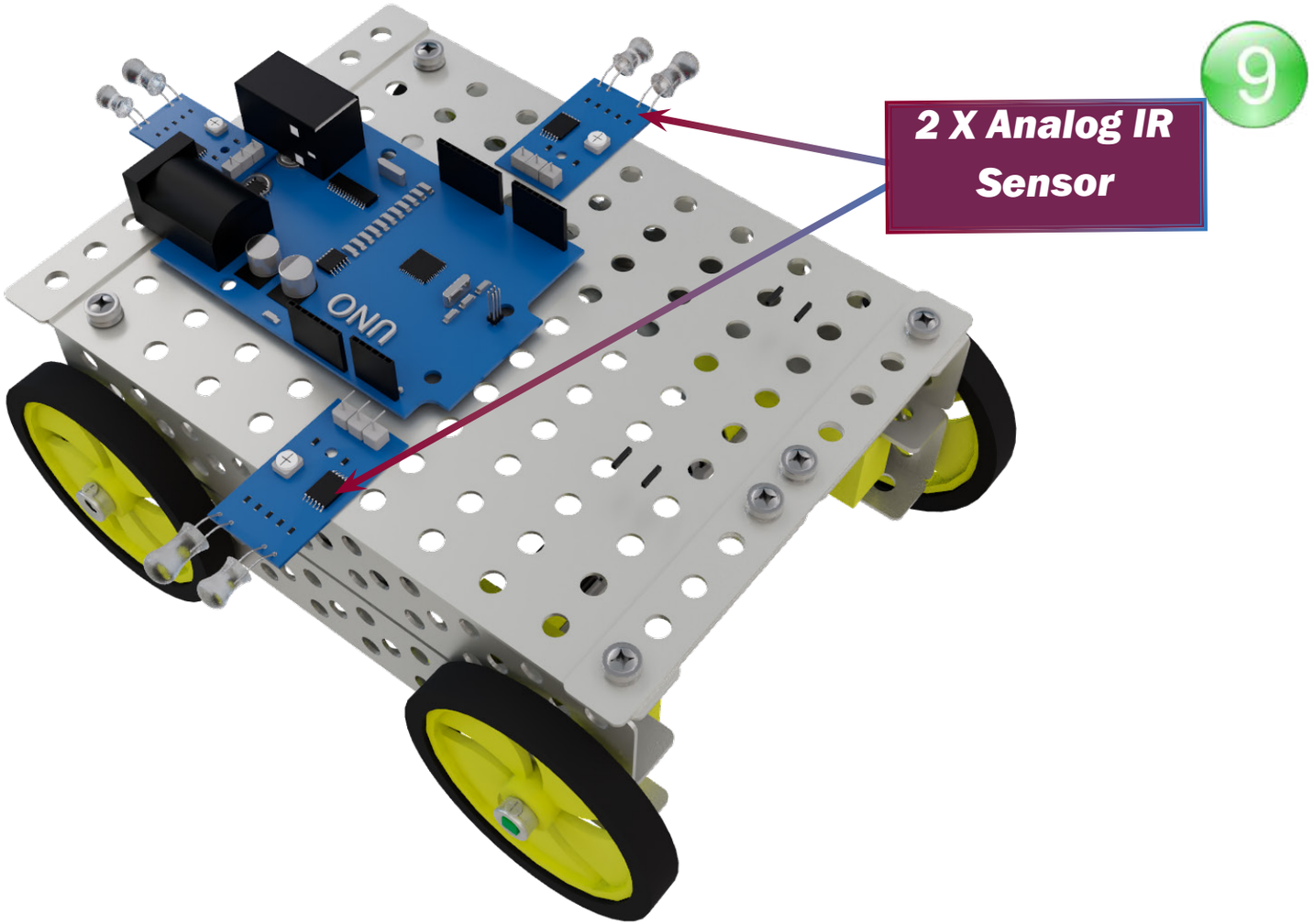


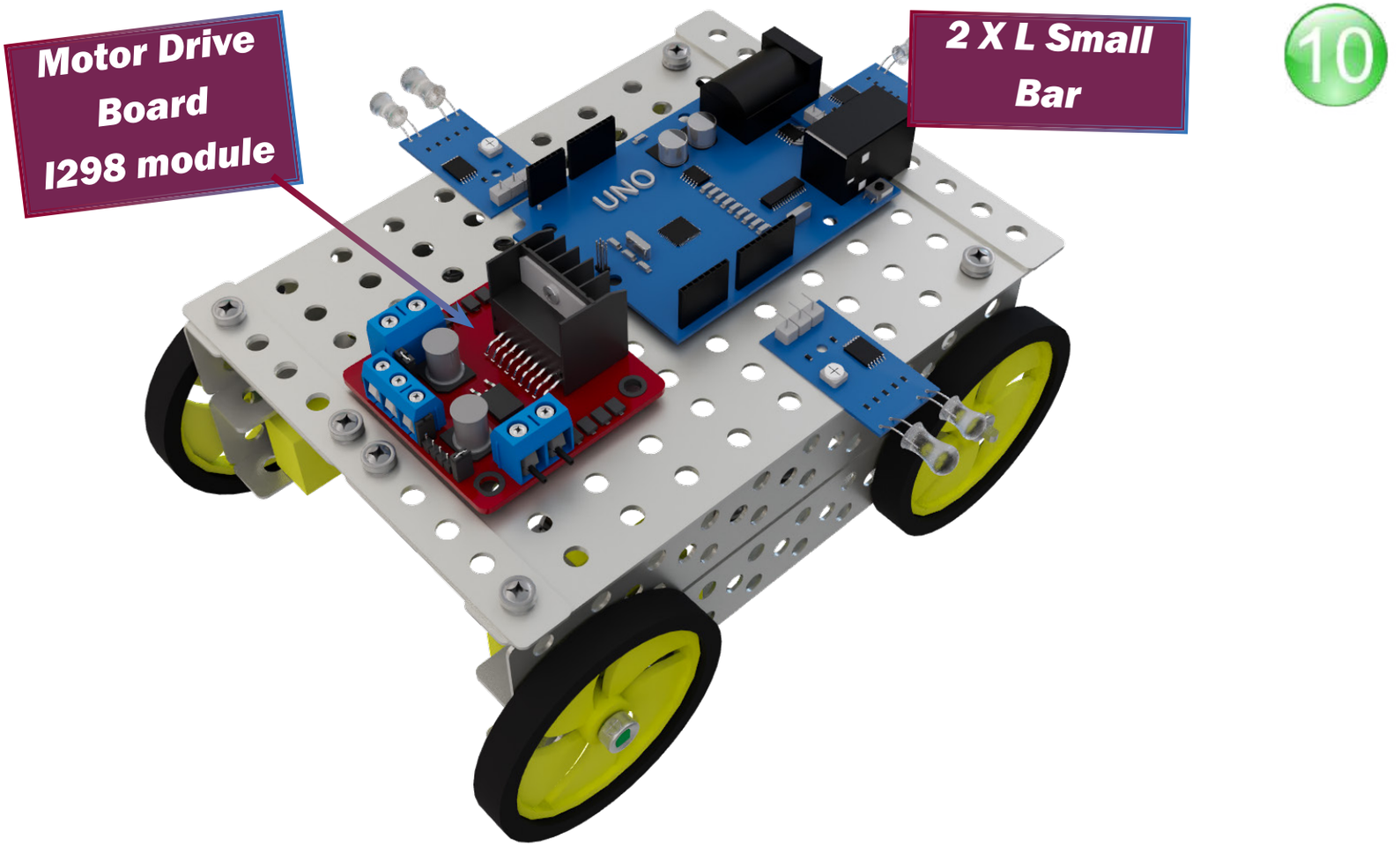




7

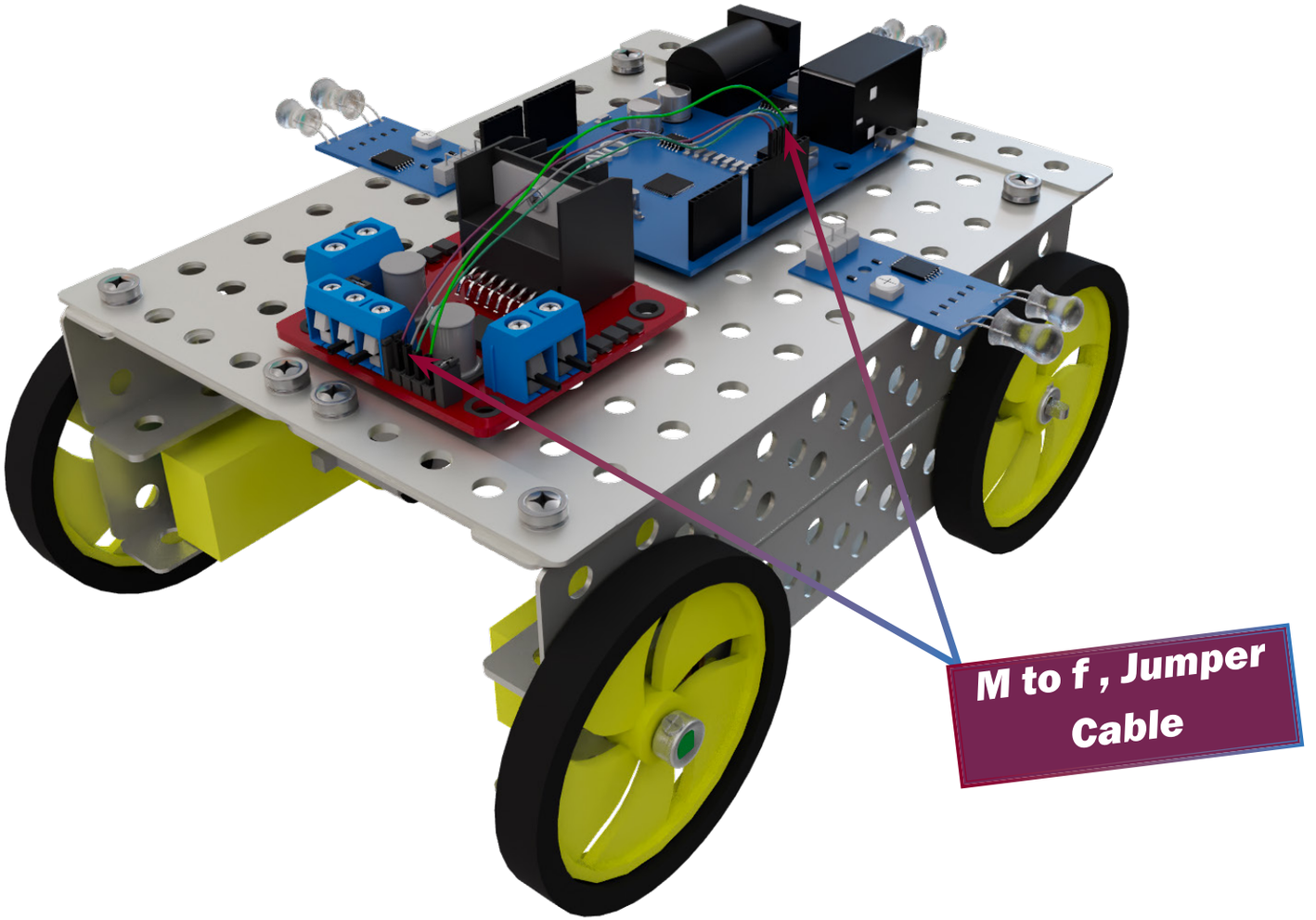


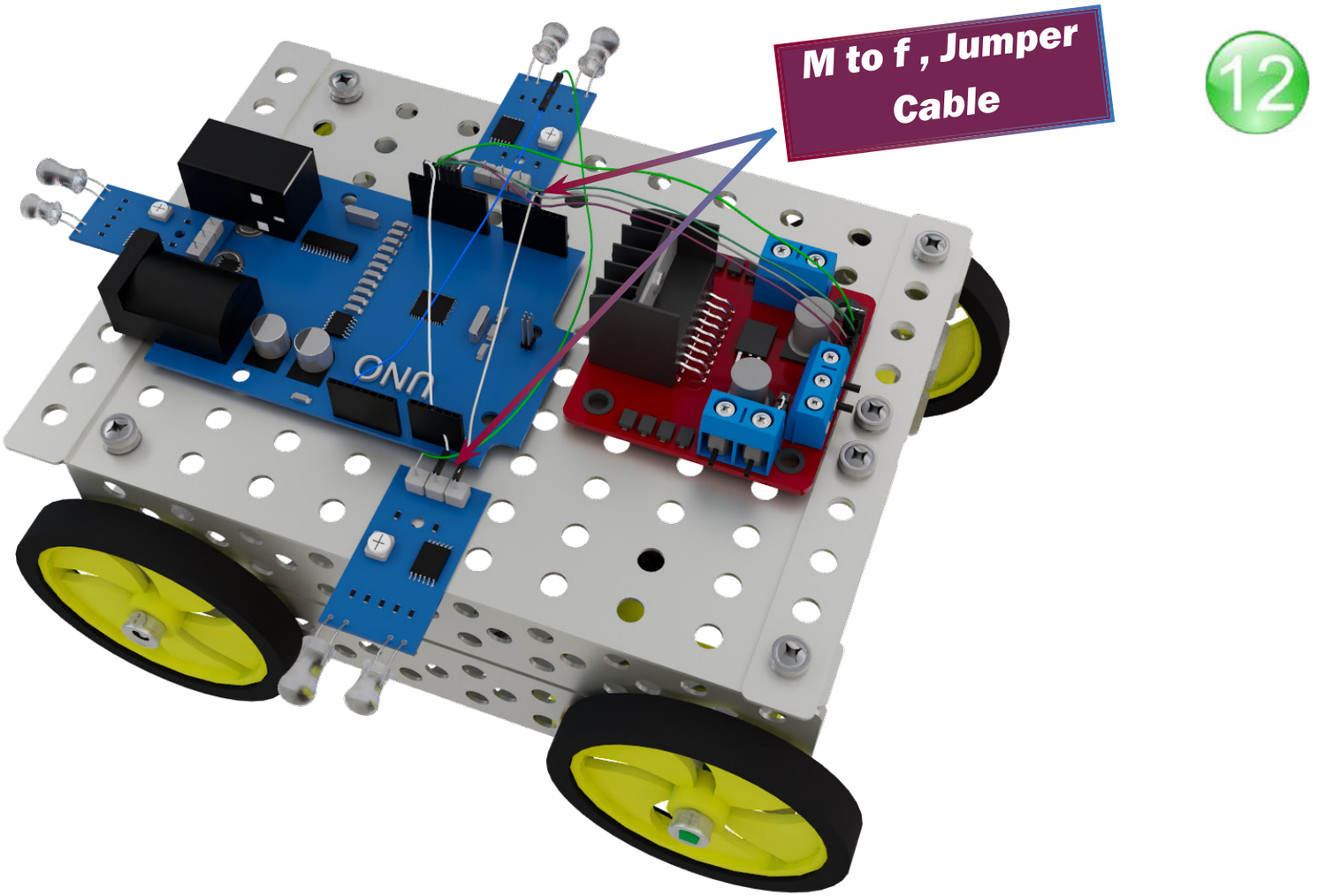




MECHANZ™

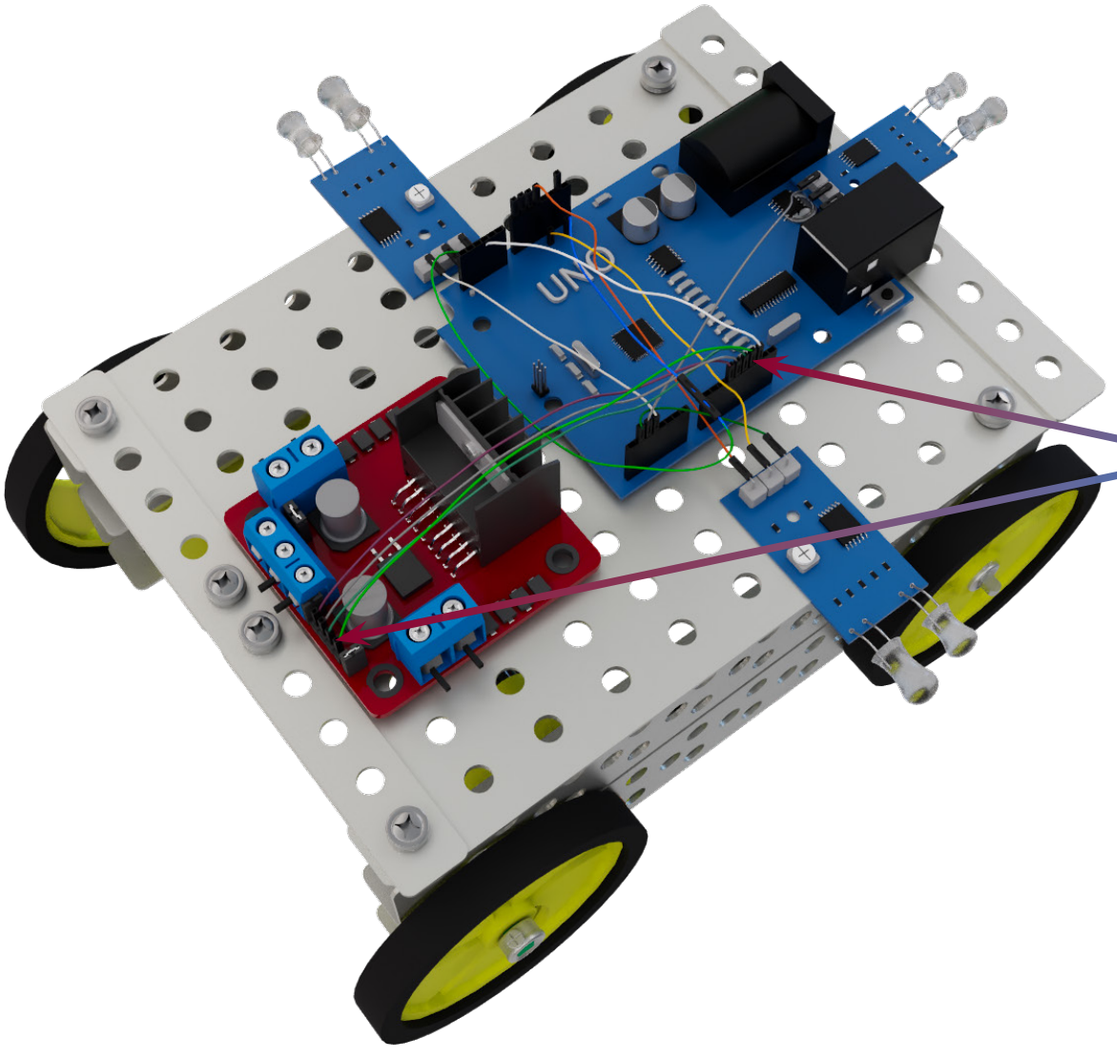
11



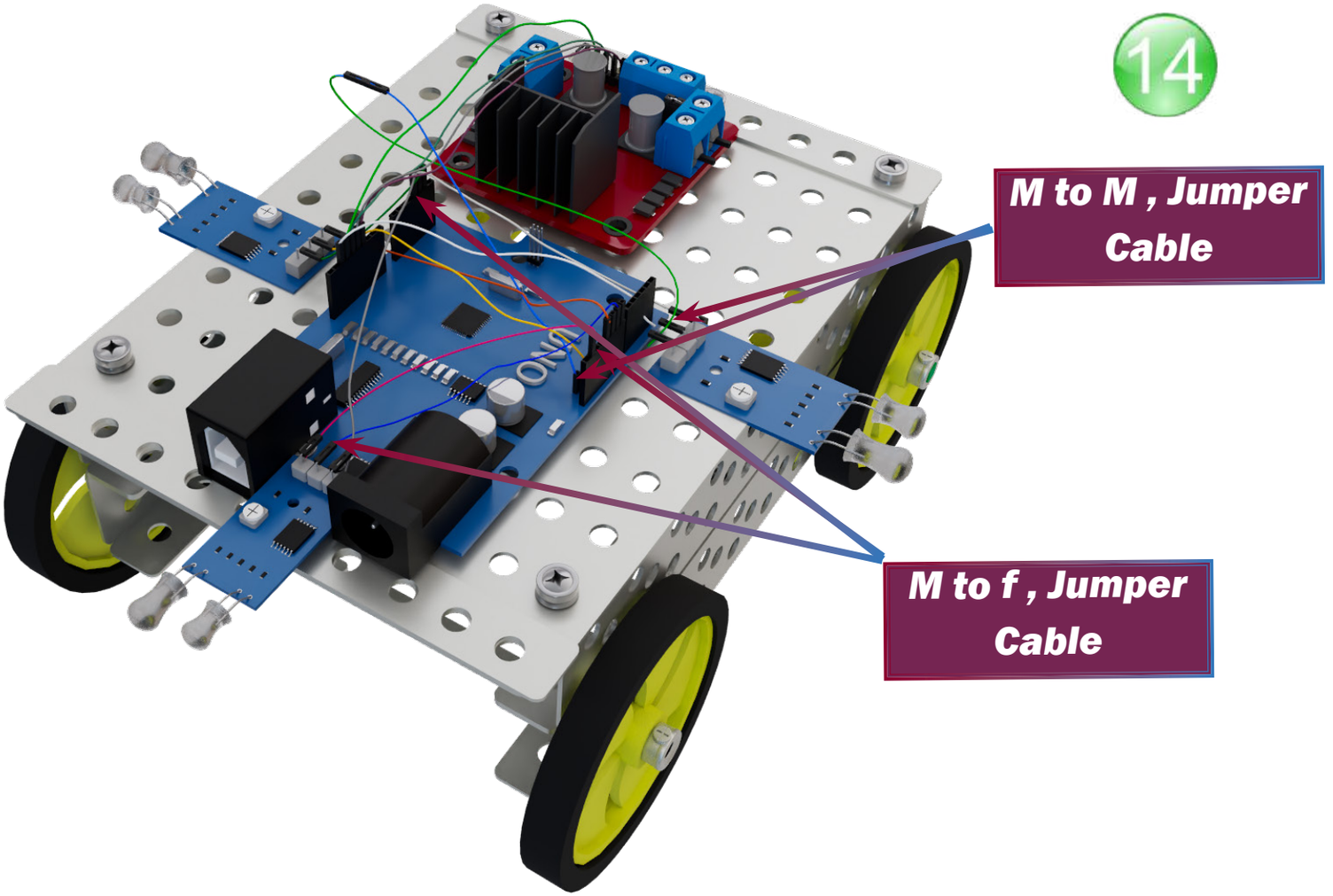


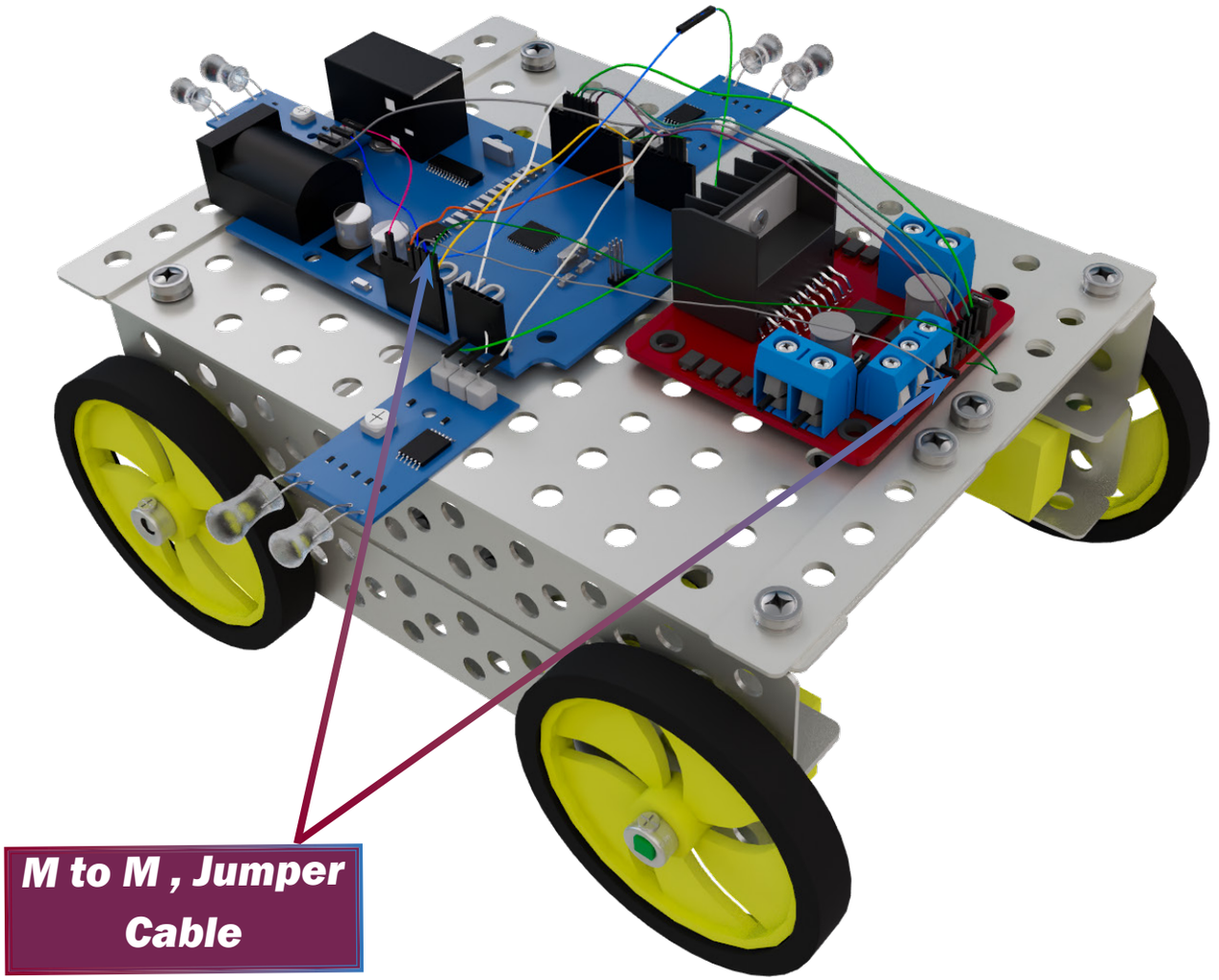
MECHANZ™

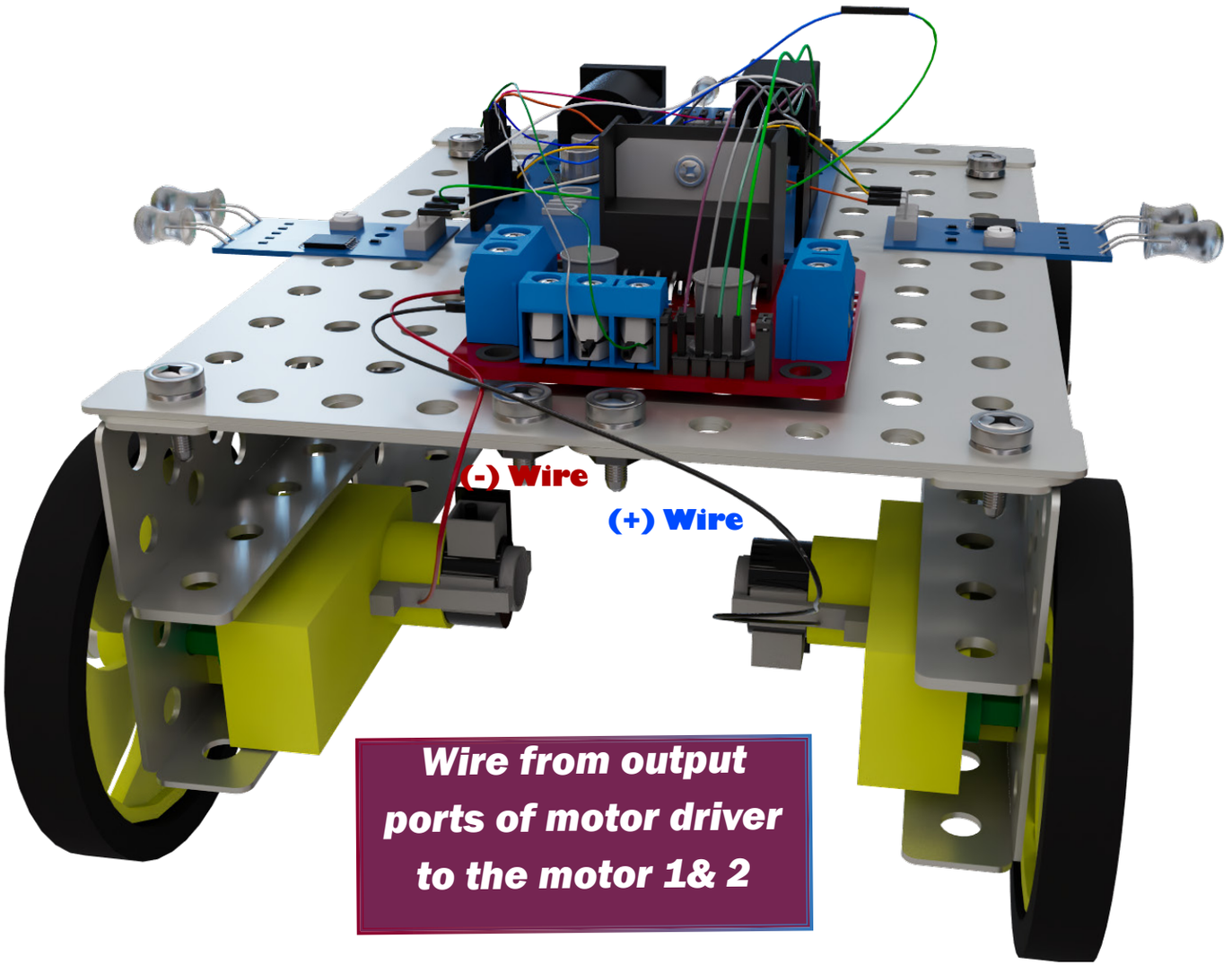
13



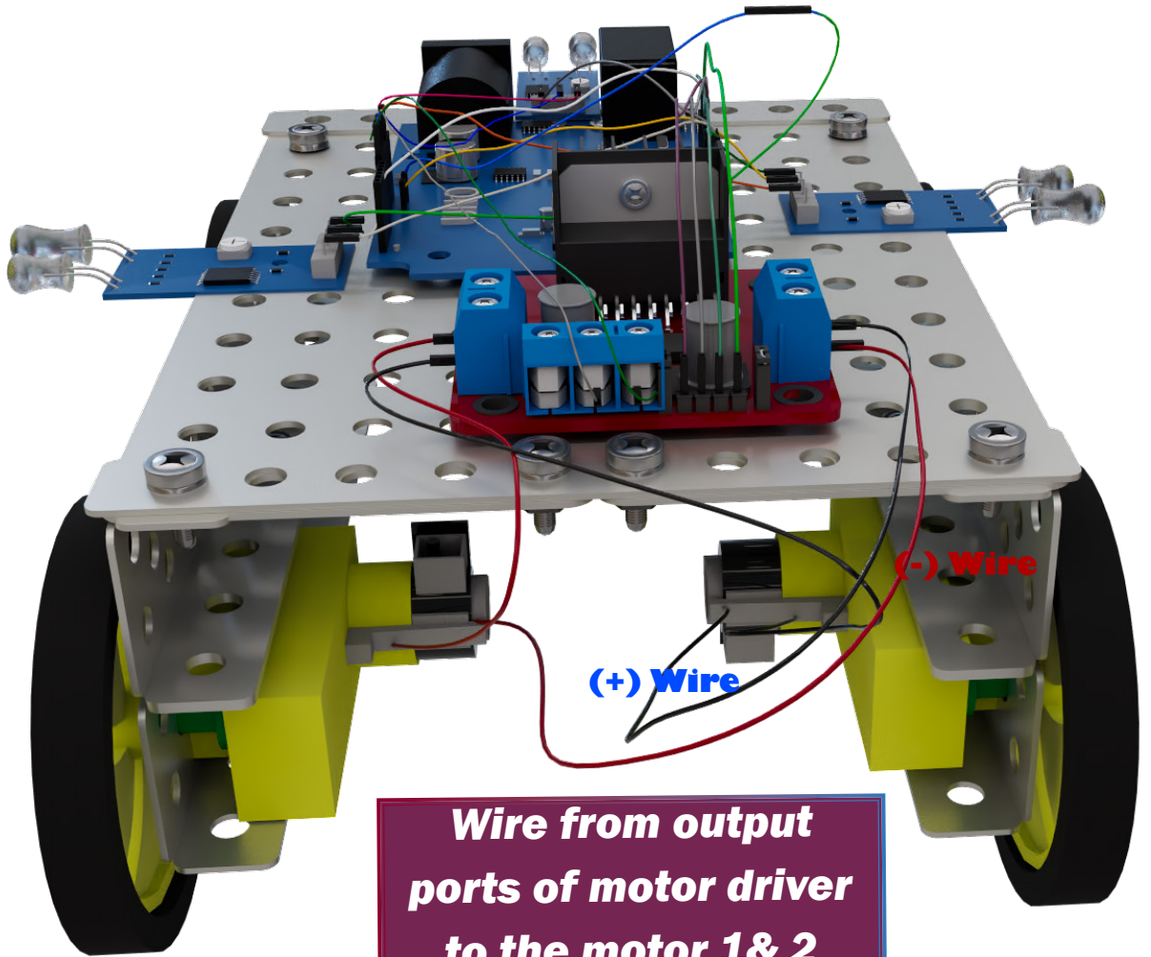
M to f , Jumper Cable







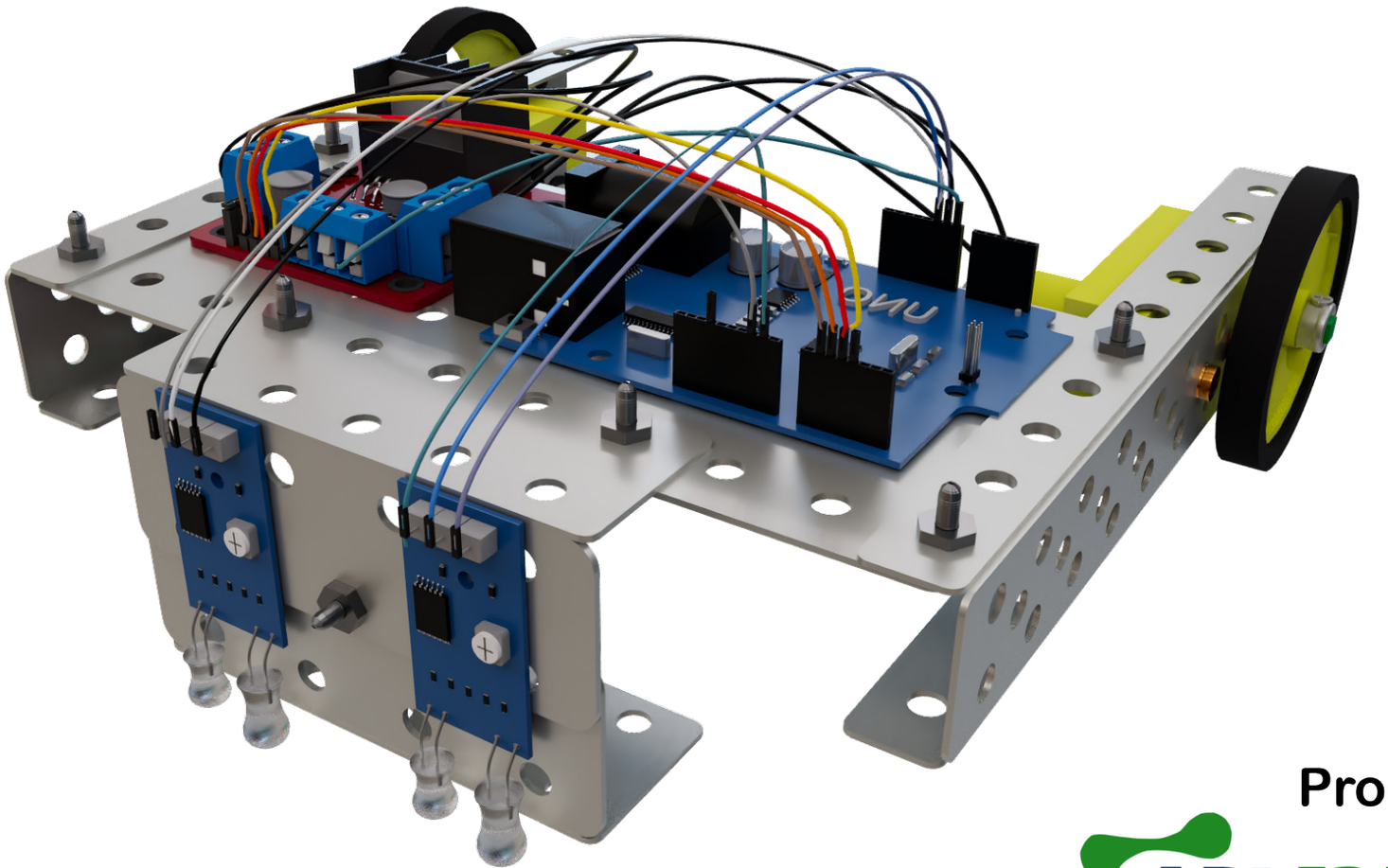
Wire from output ports of motor driver to the motor 1& 2



Wire from output ports of motor driver to the motor 1& 2

MECHANZ™

Line Following Bot

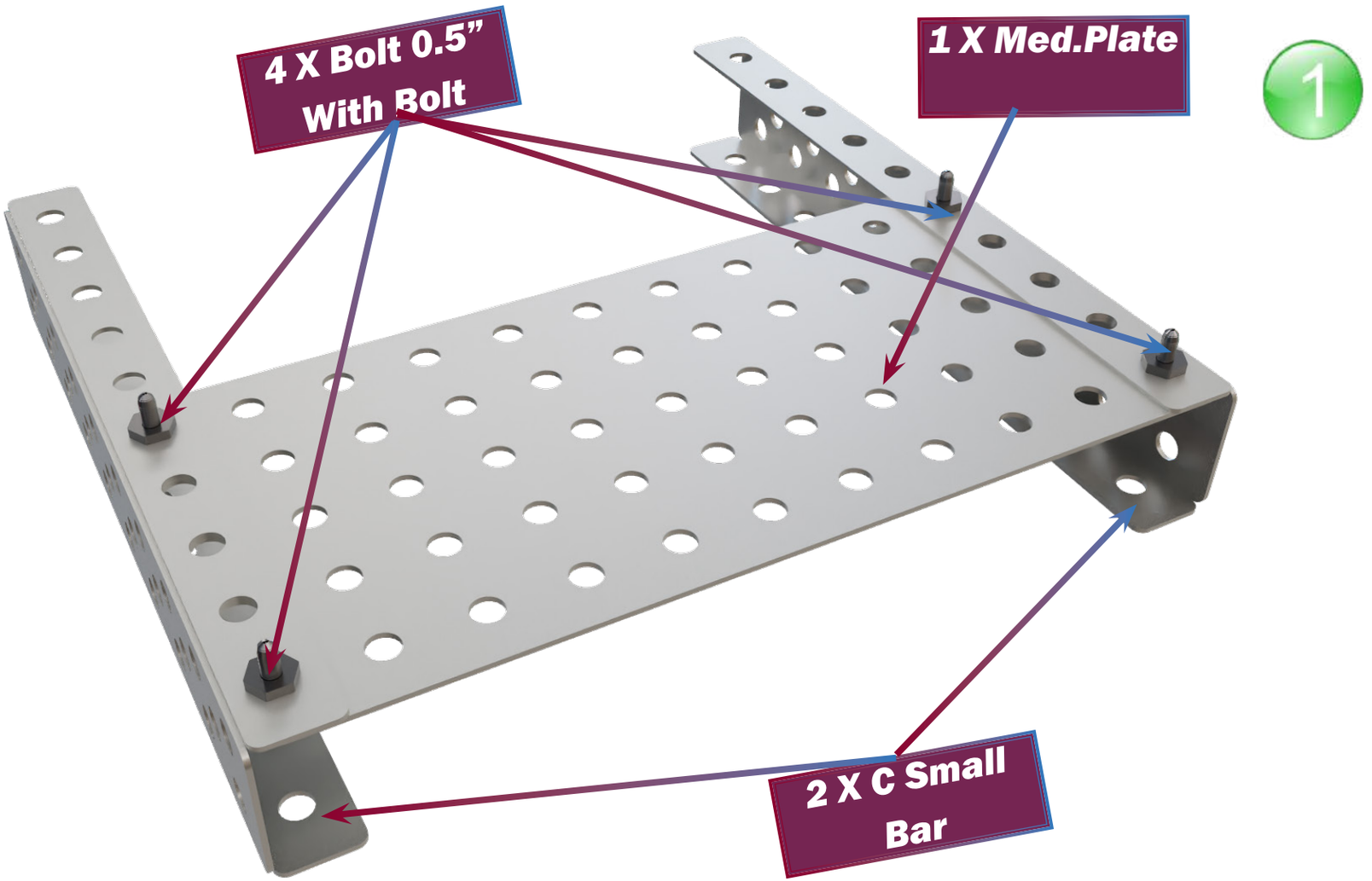


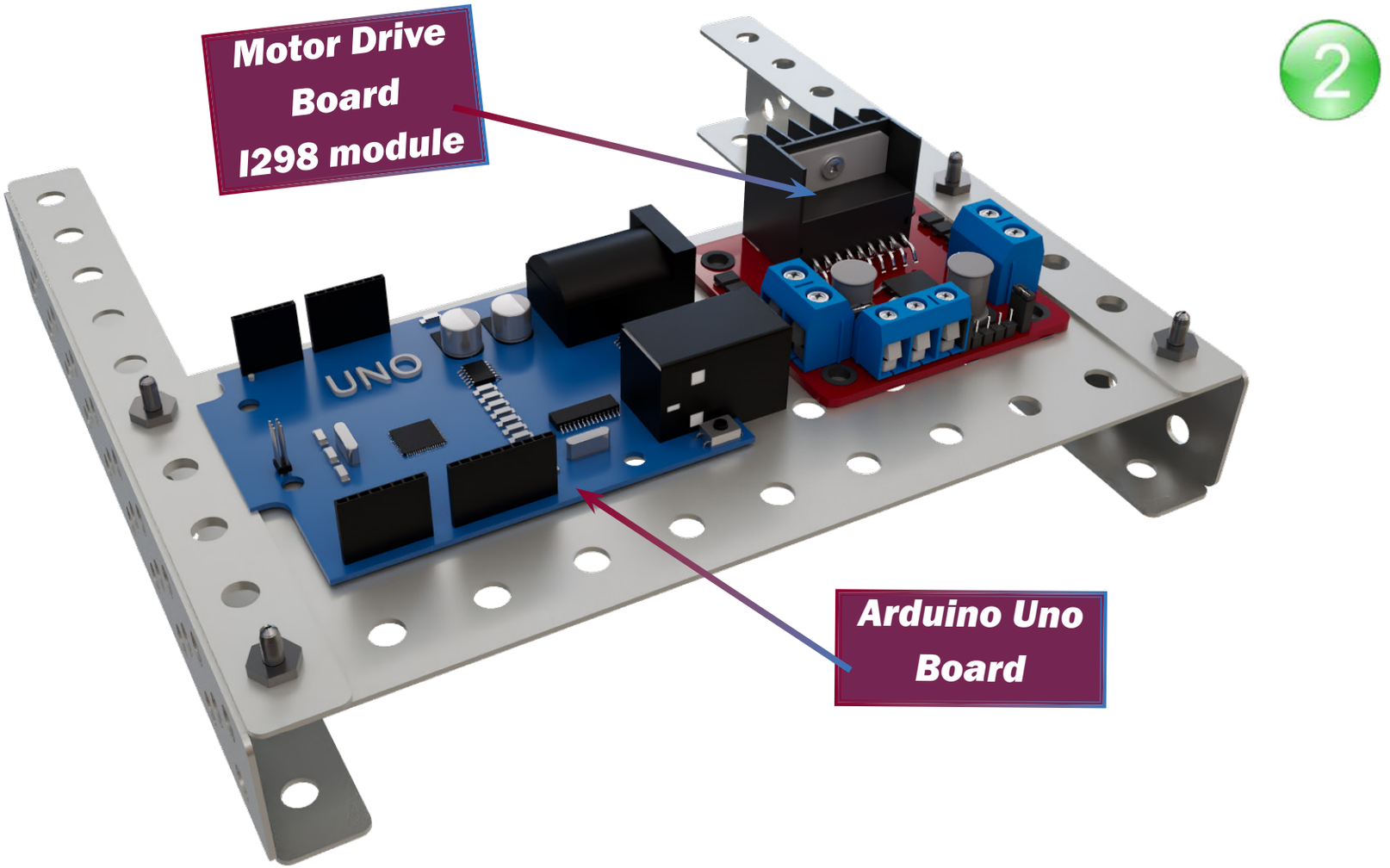
Product of

ABLEDUCATION™

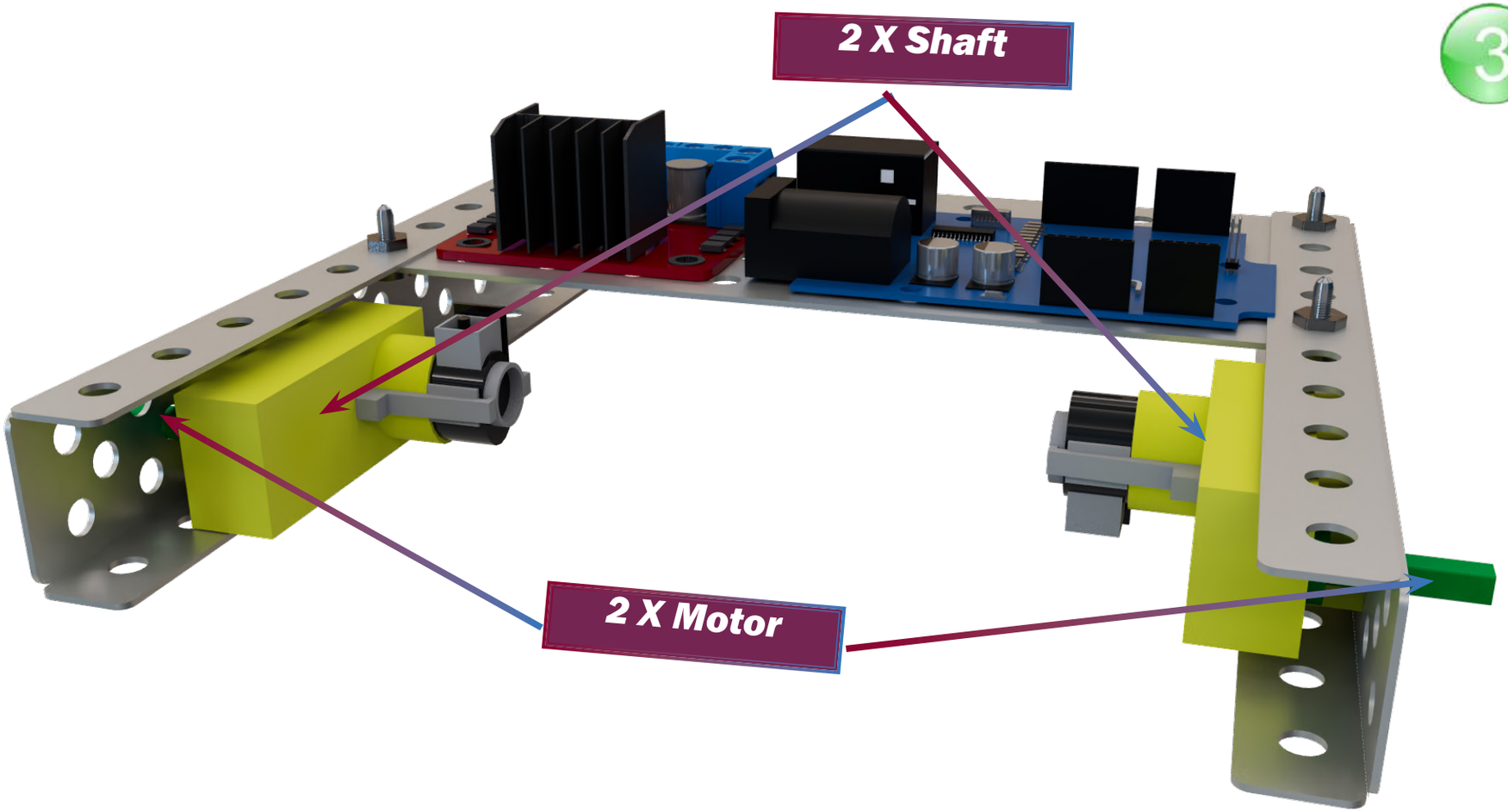
THINK • EXPLORE • CREATE • INNOVATE

MECHANZ™

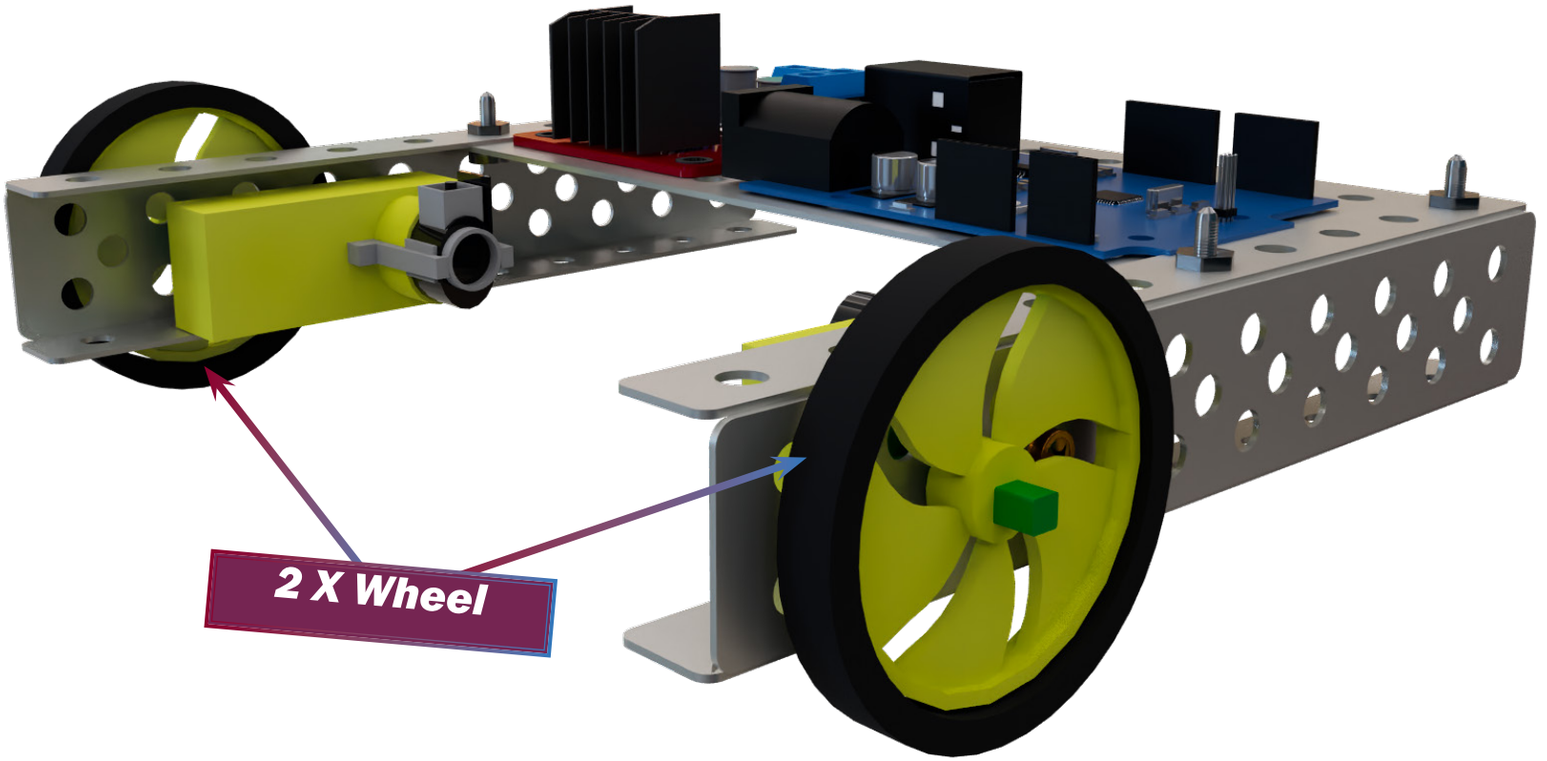




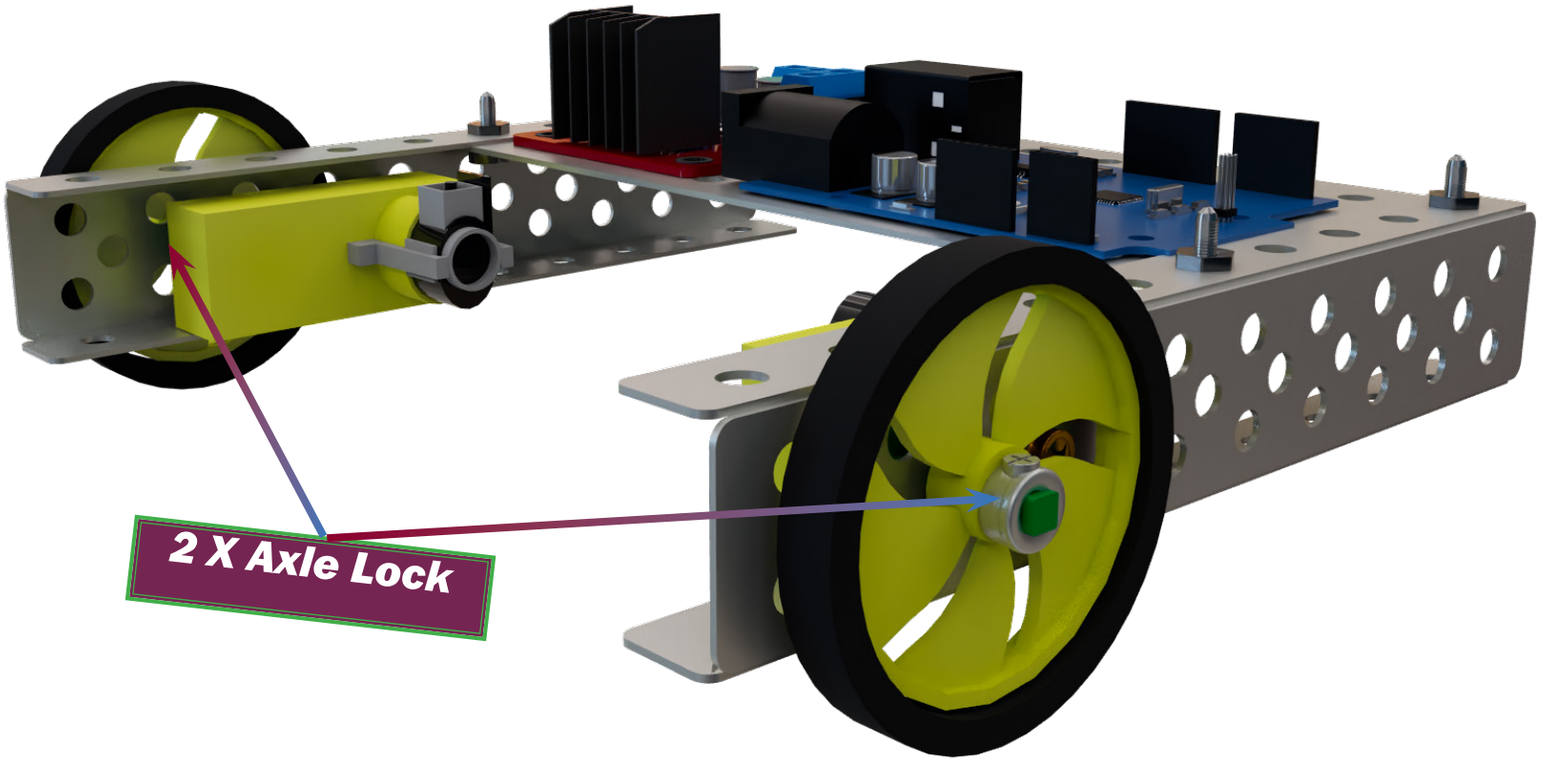
2



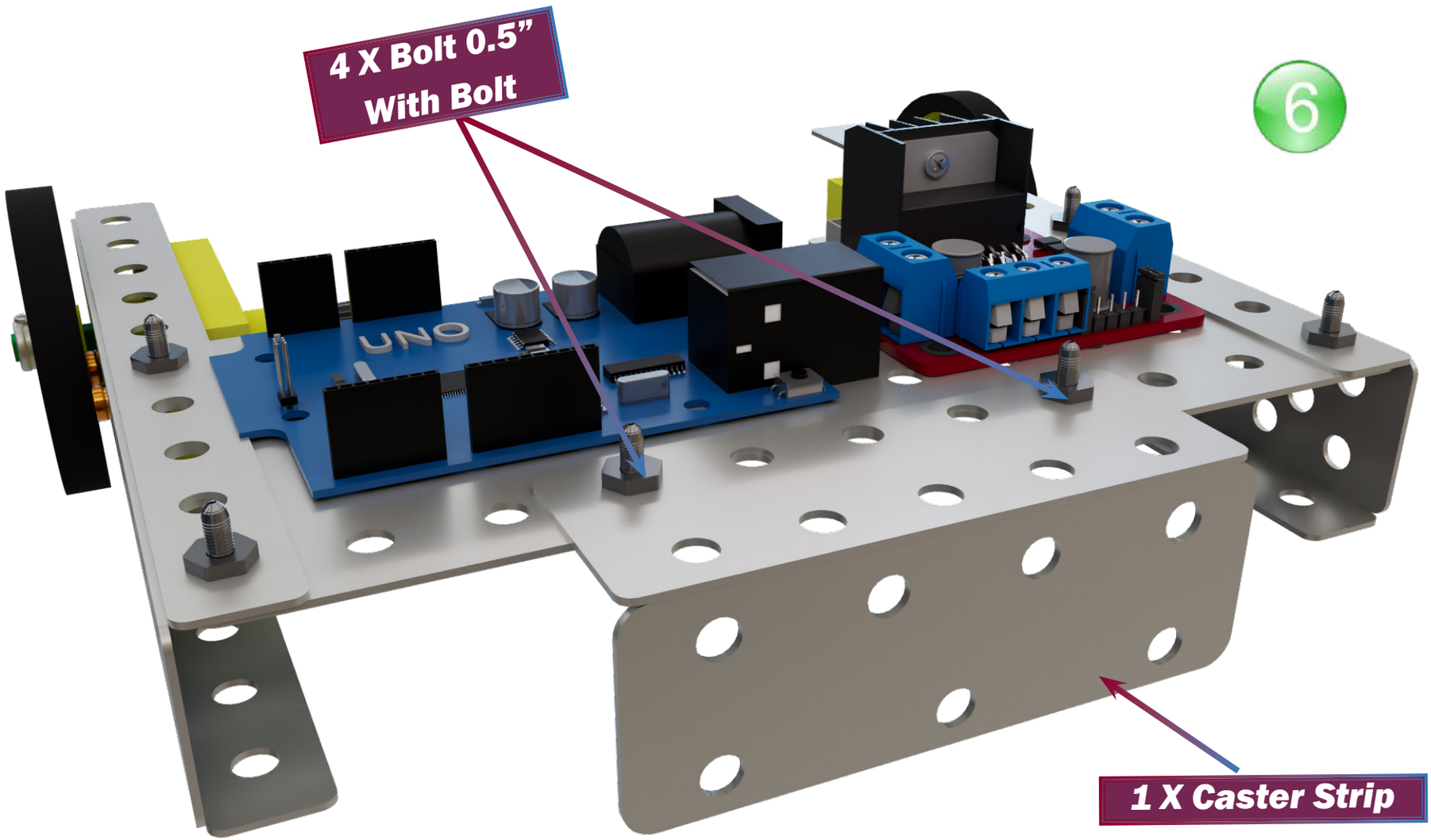
4



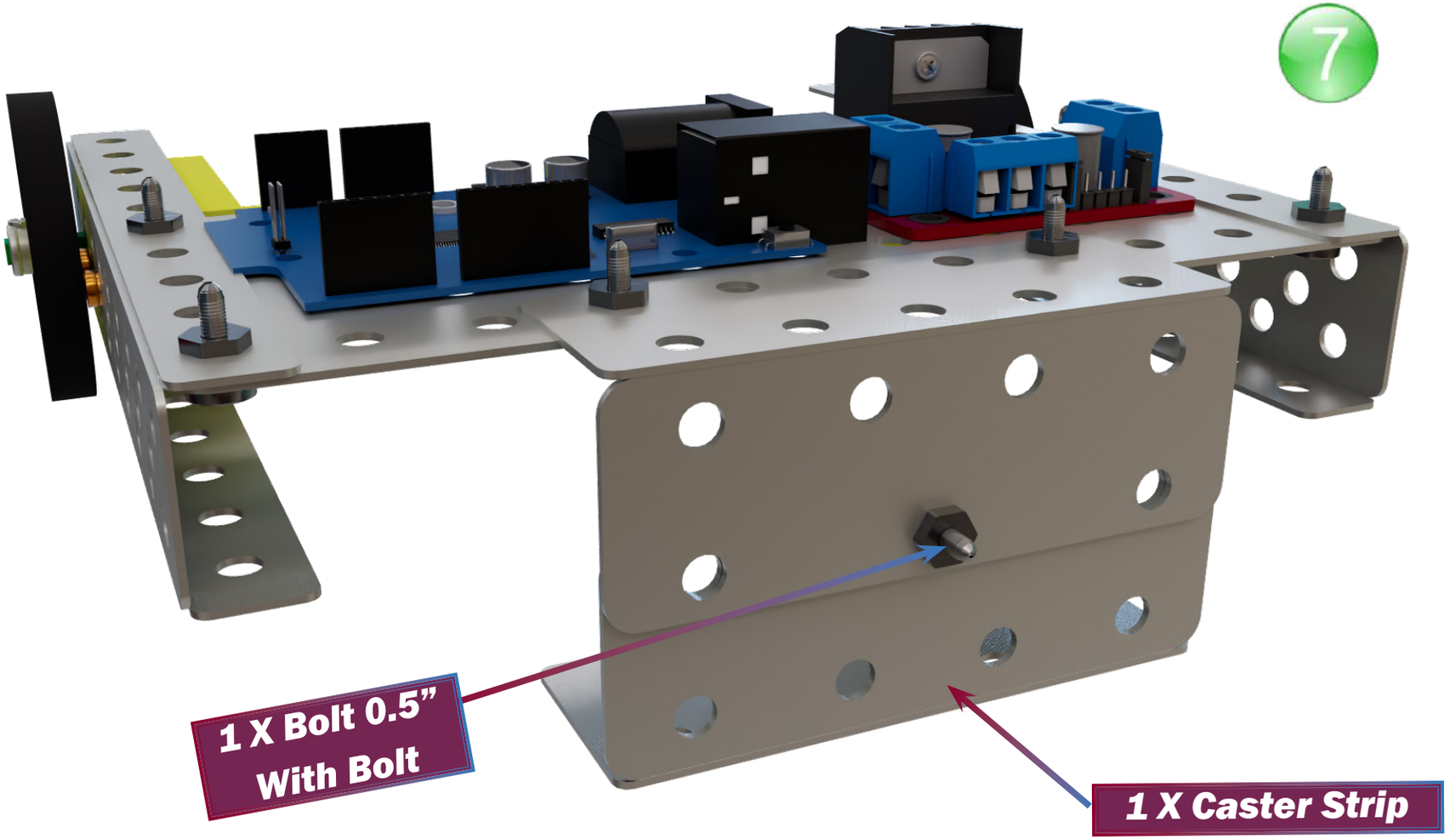
5

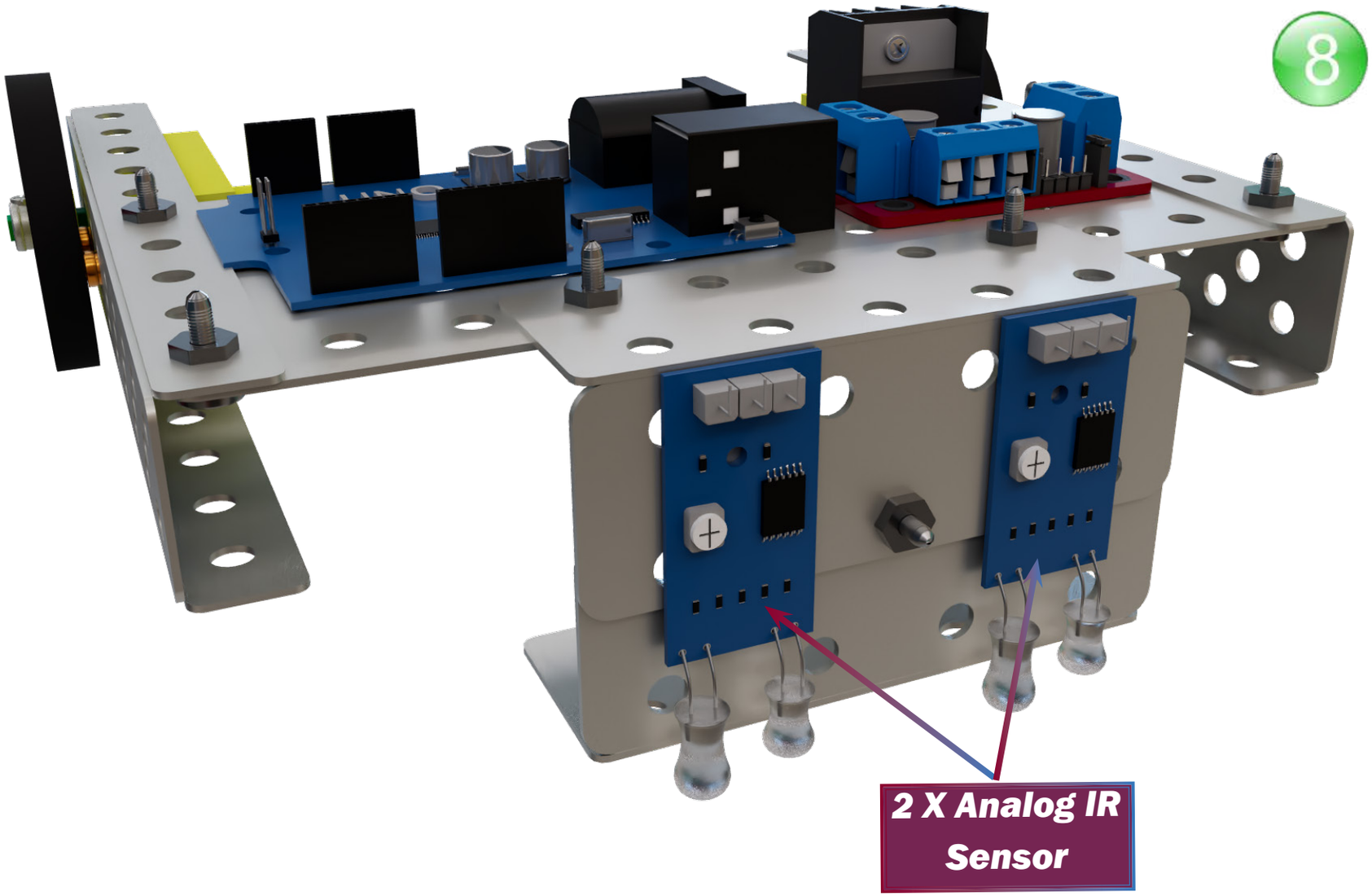


MECHANZ™

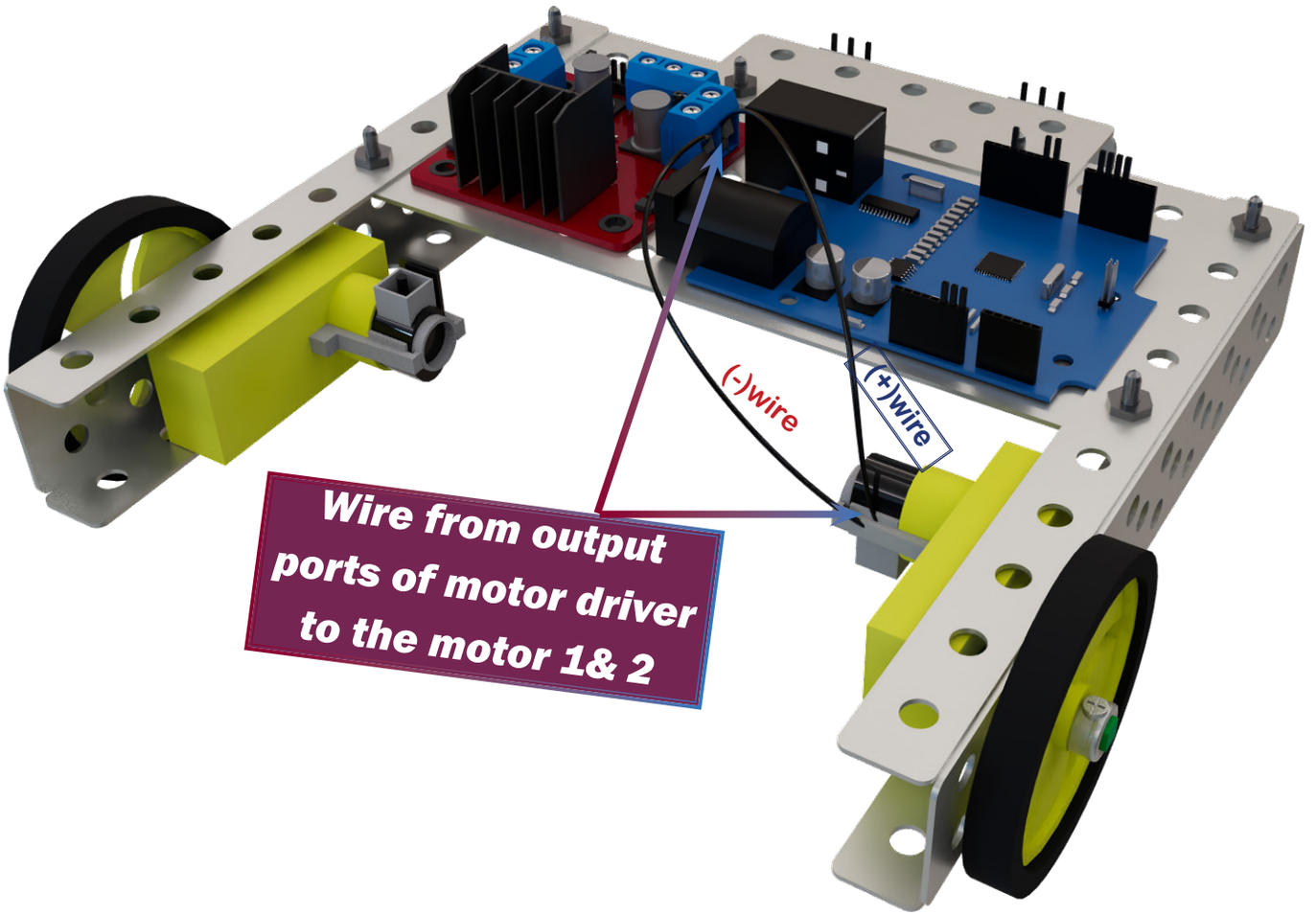


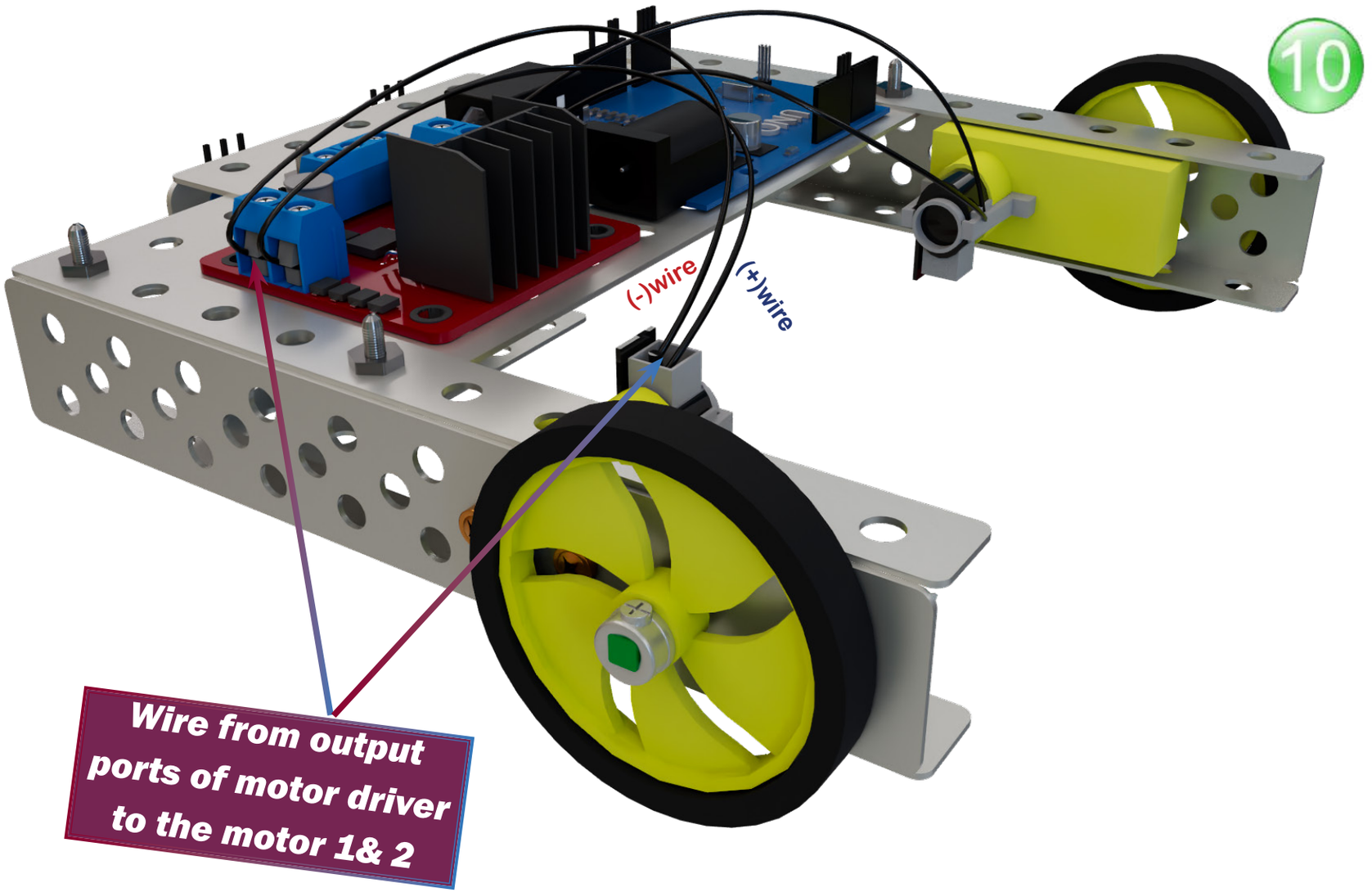
MECHANZ™

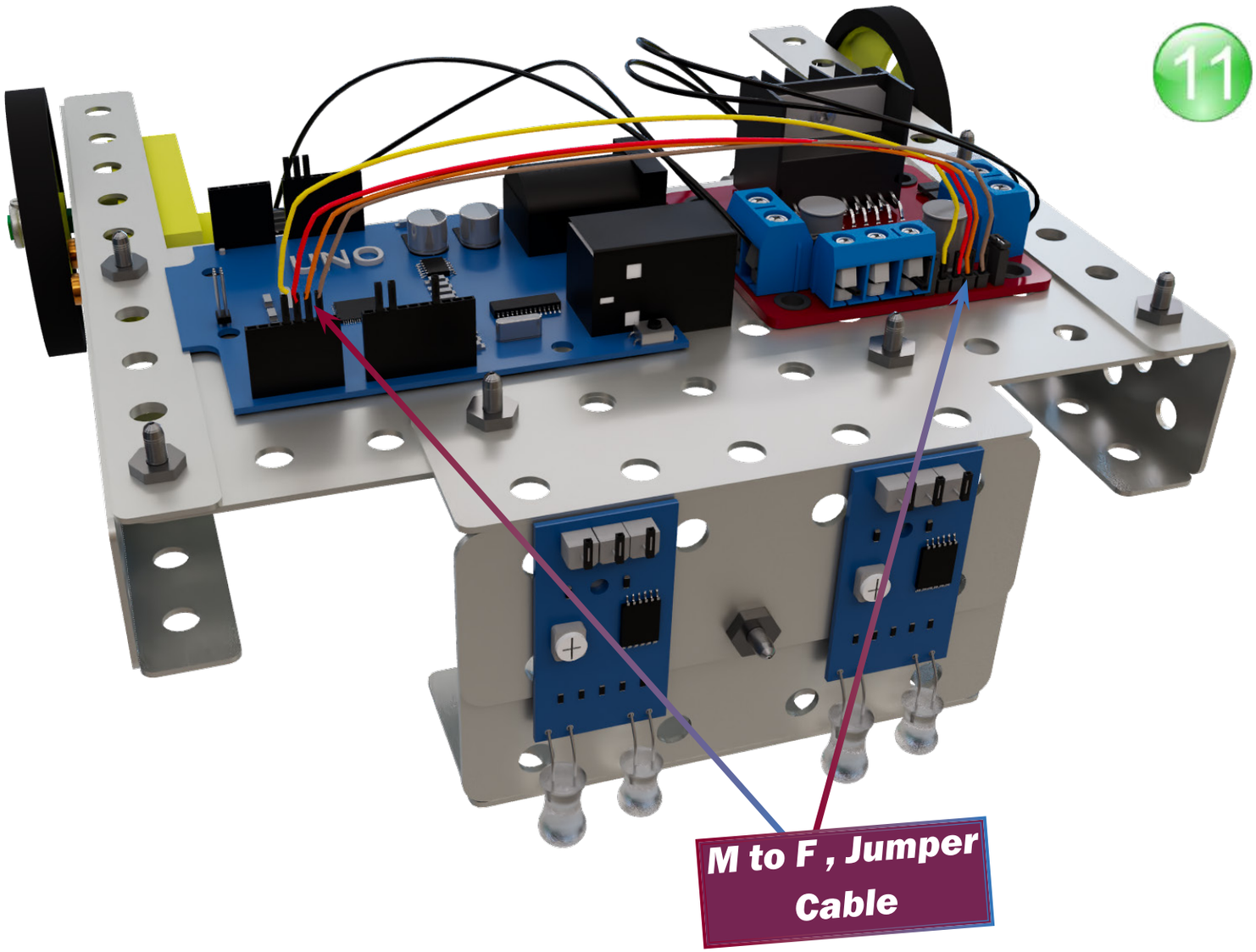




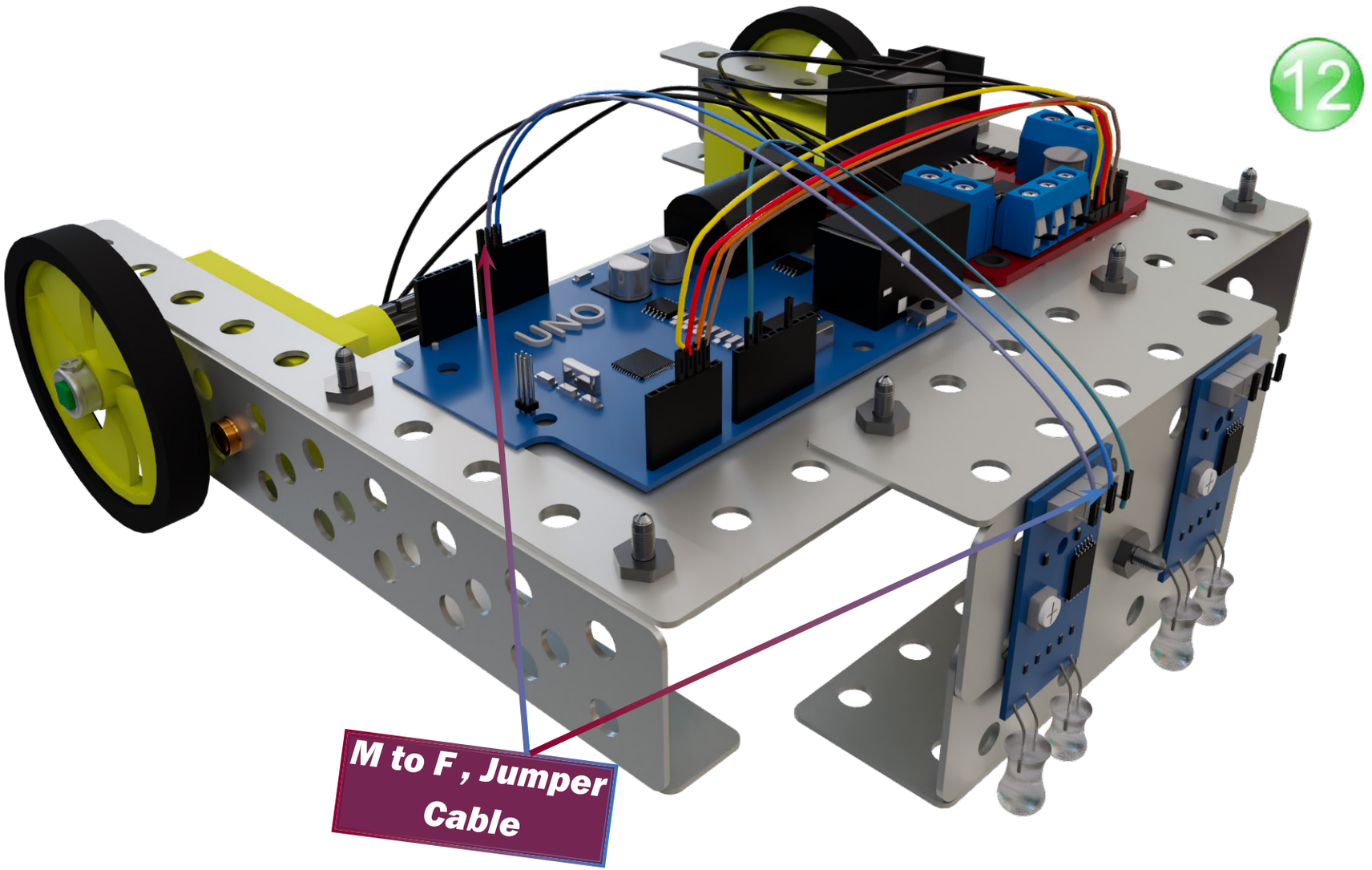
9

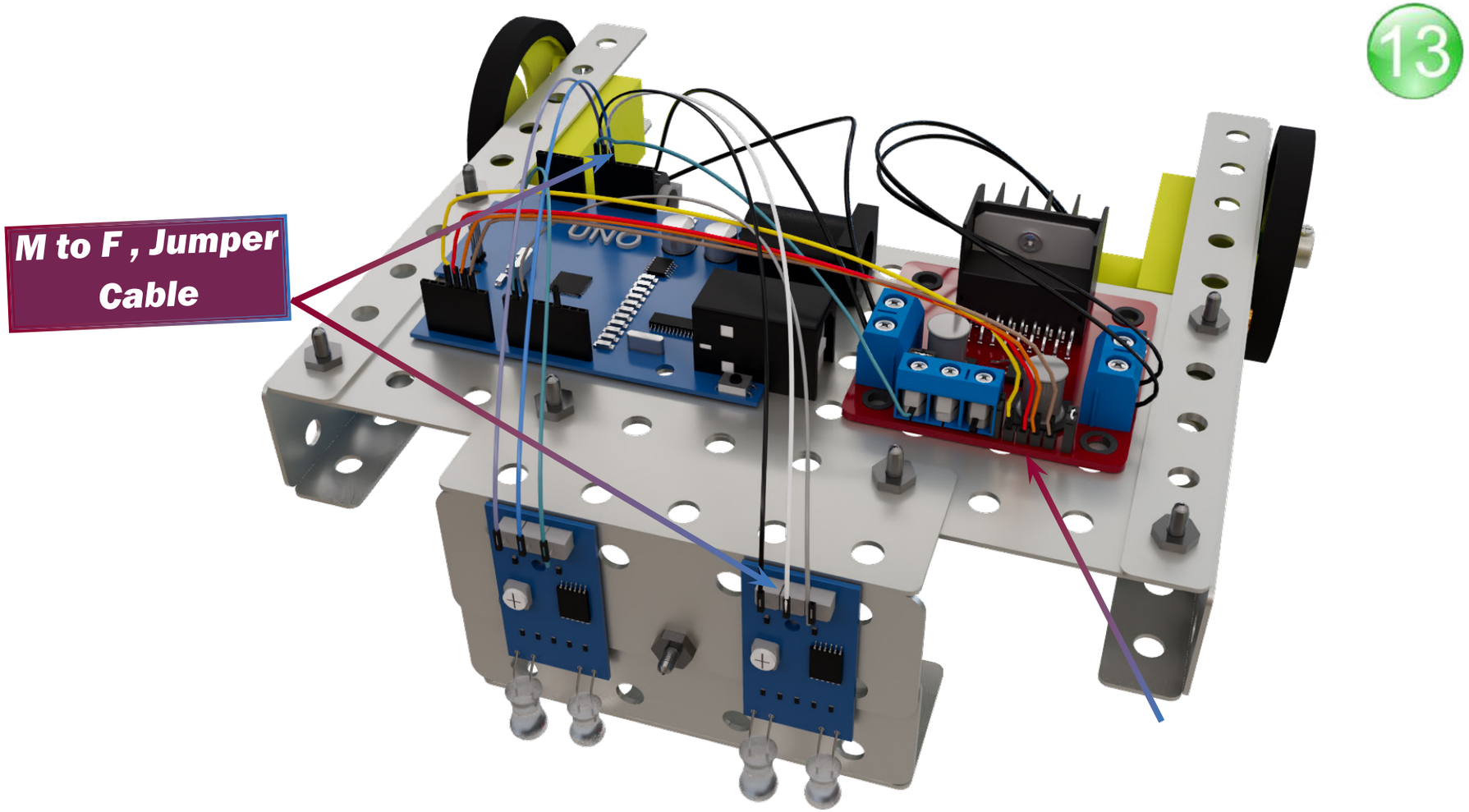






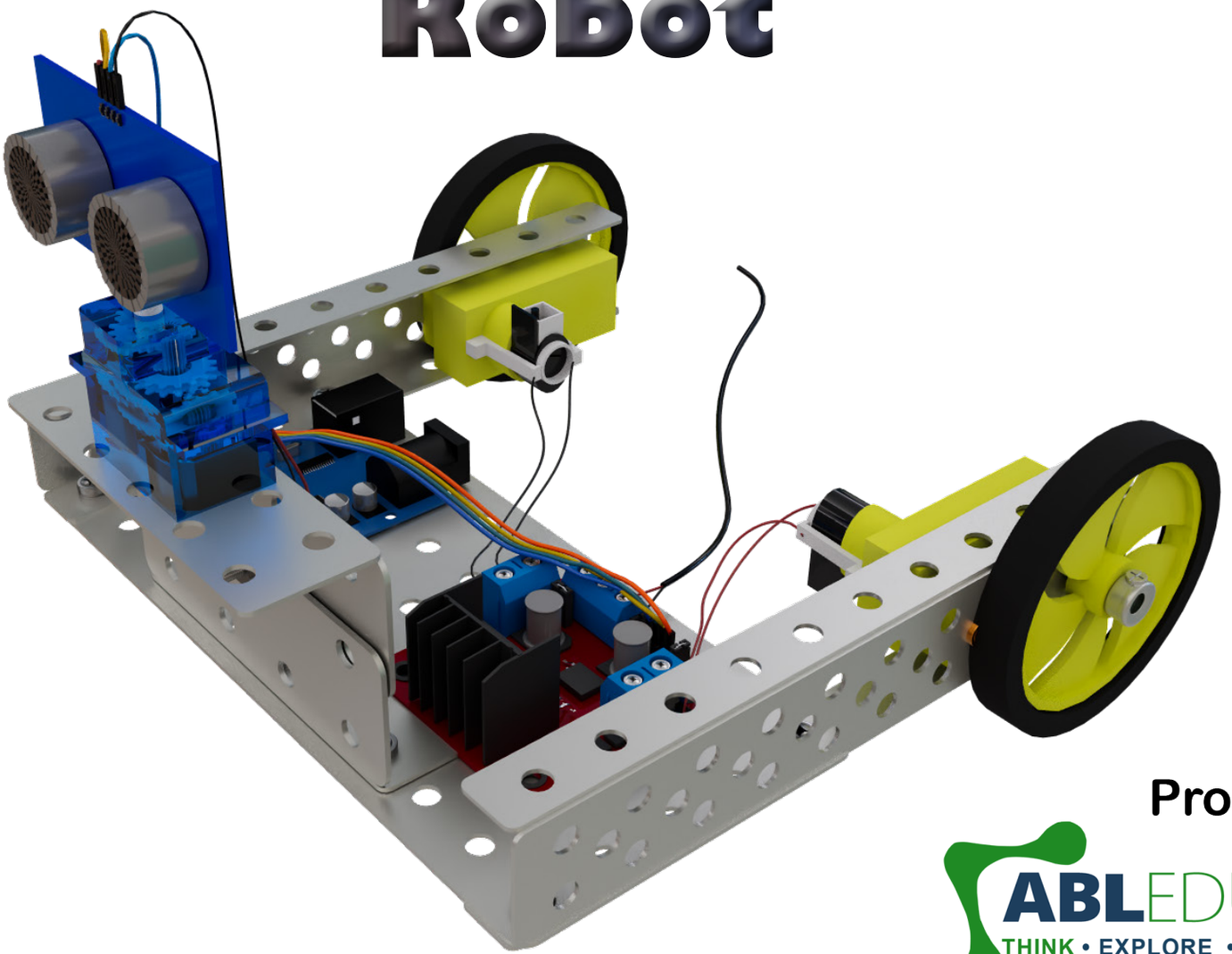
MECHANZ™





MECHANZ™

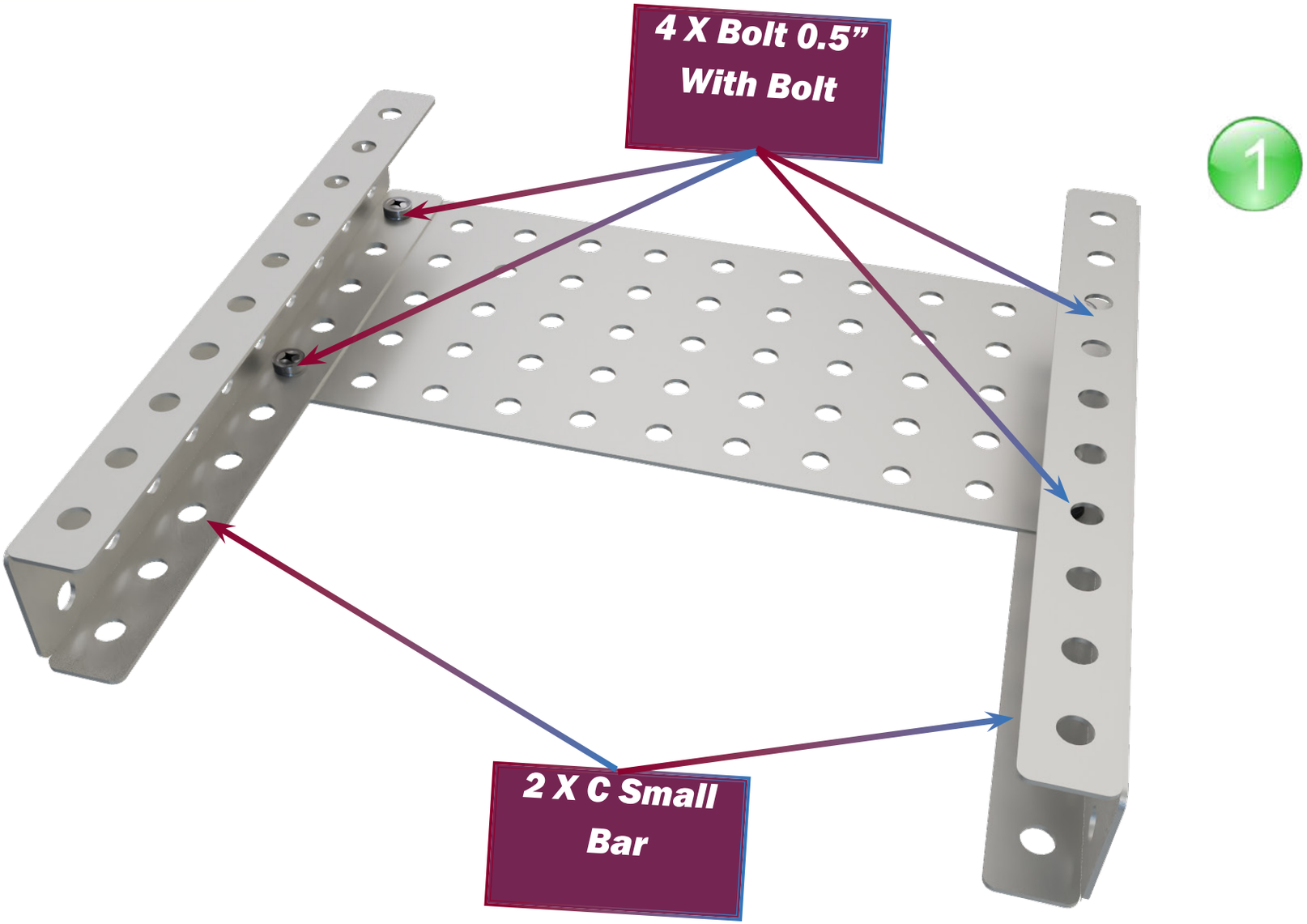
Obstacle Avoiding Robot



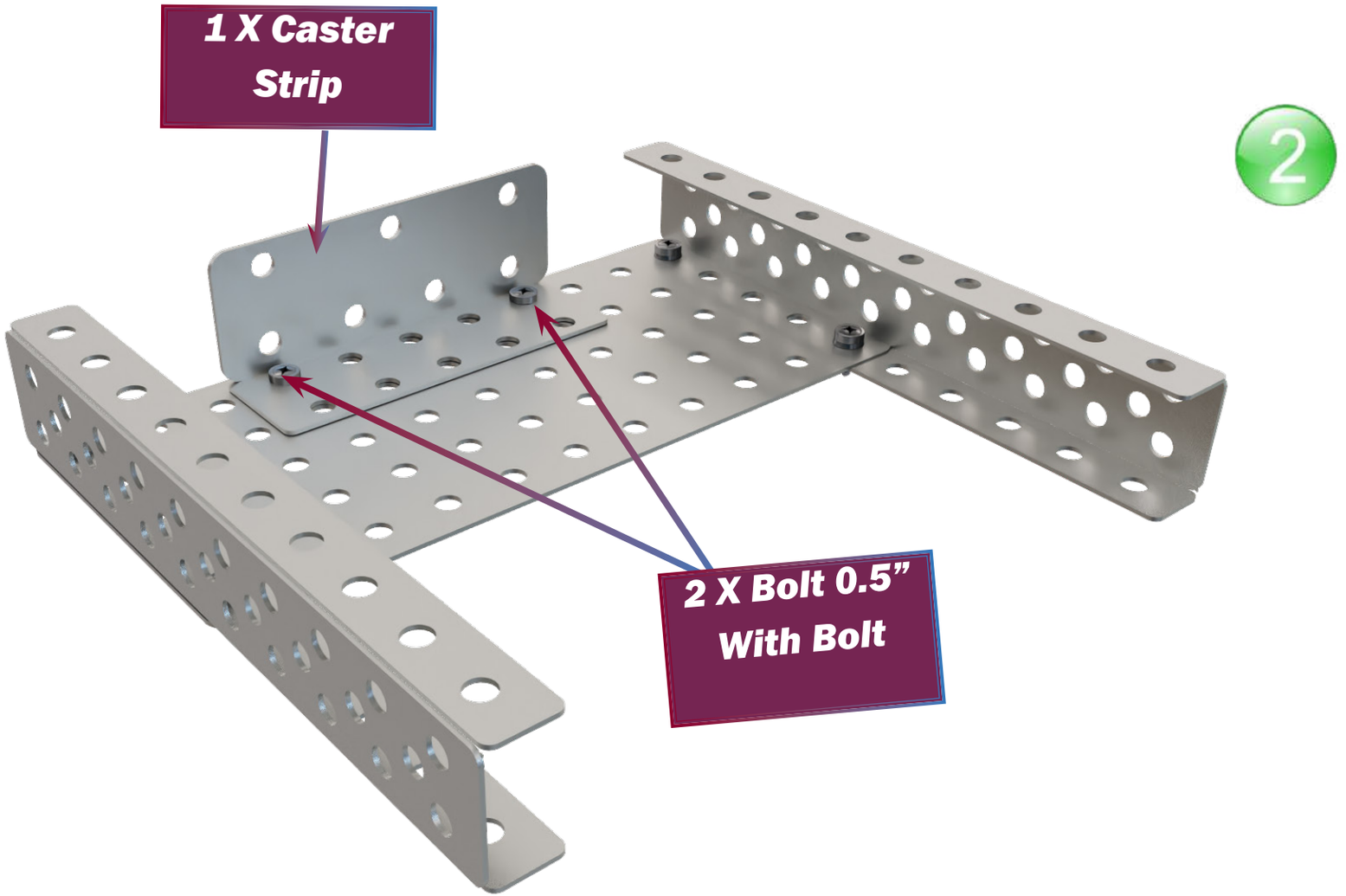
Product of

ABLEDUCATION™

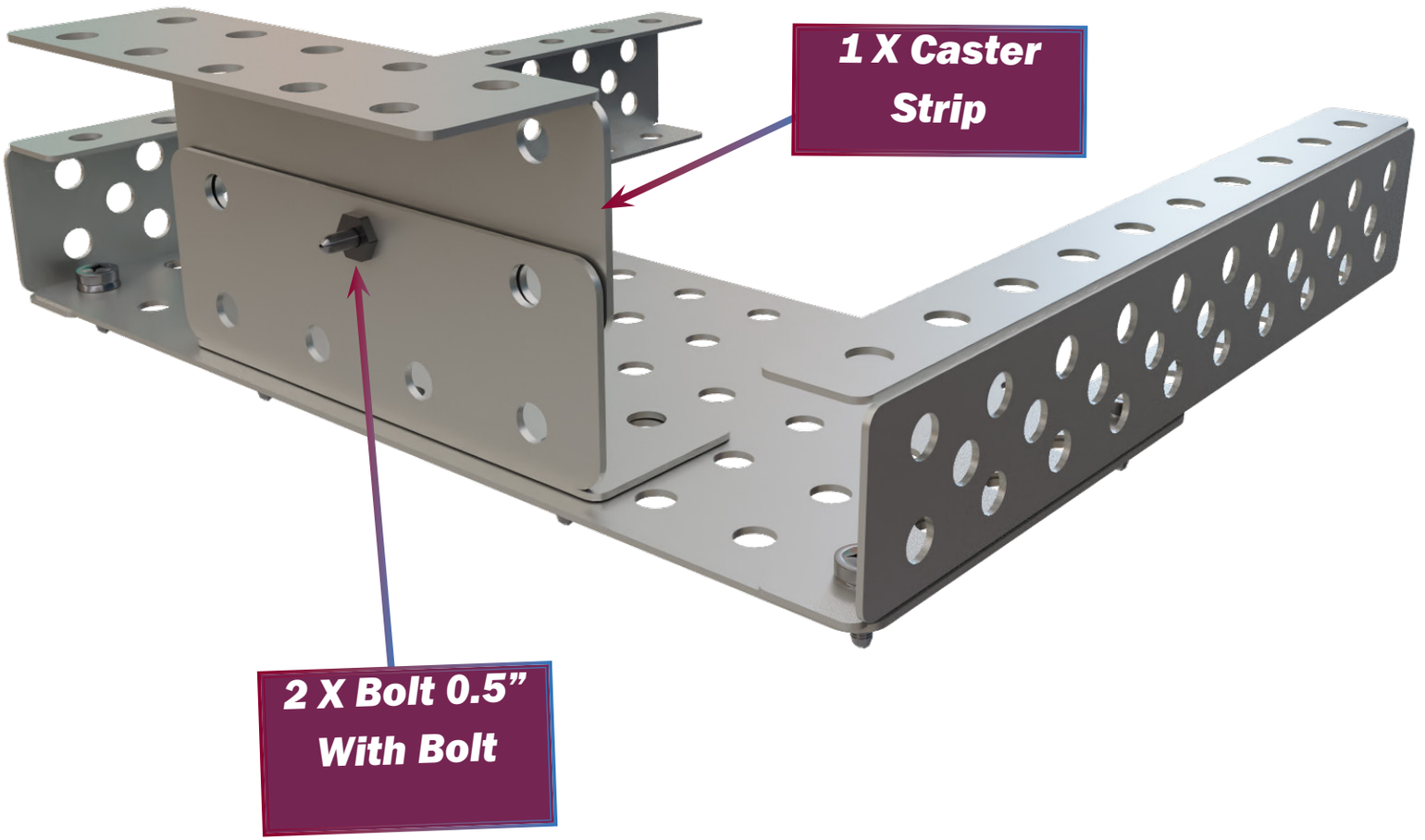
THINK • EXPLORE • CREATE • INNOVATE

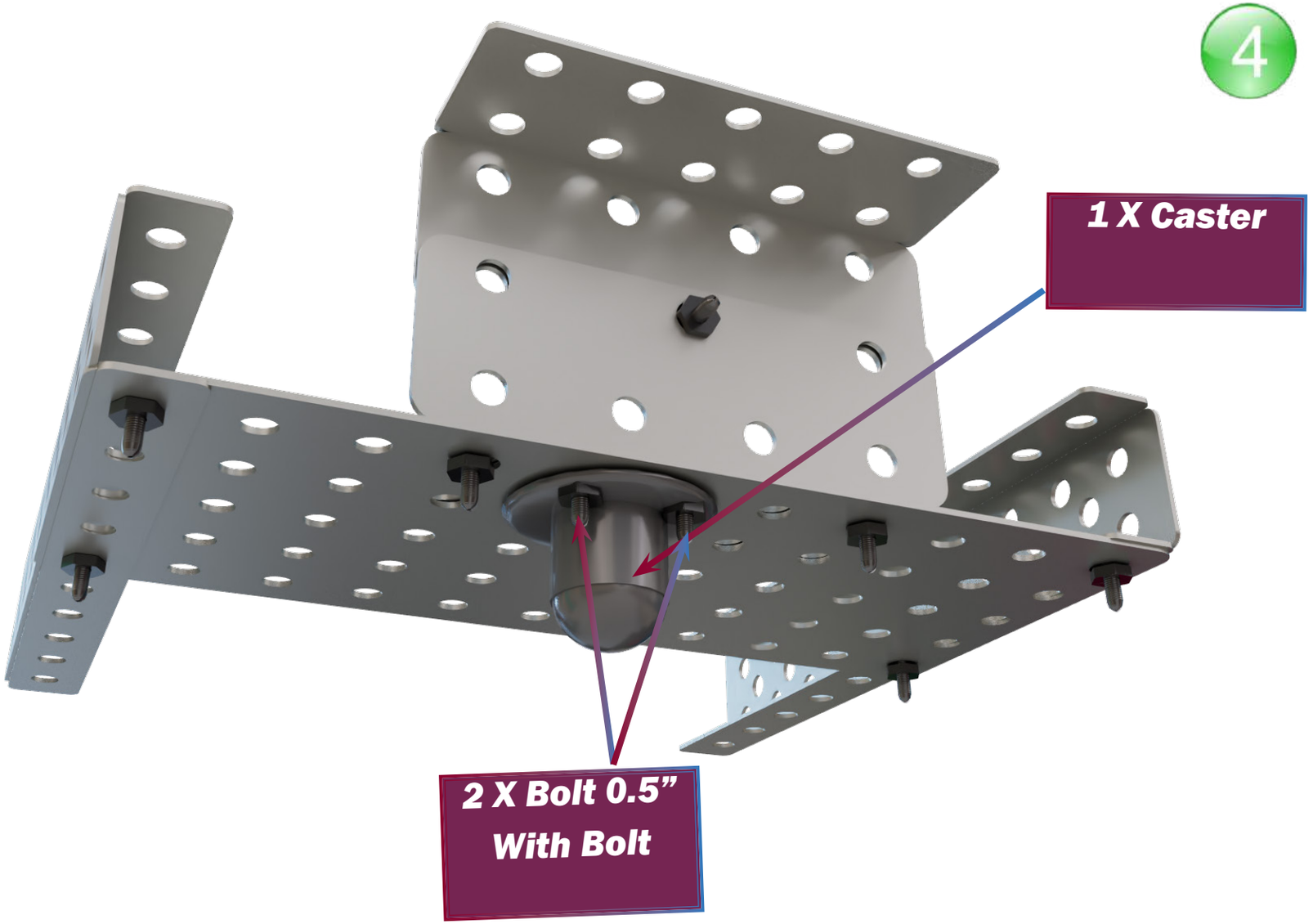


MECHANZ™

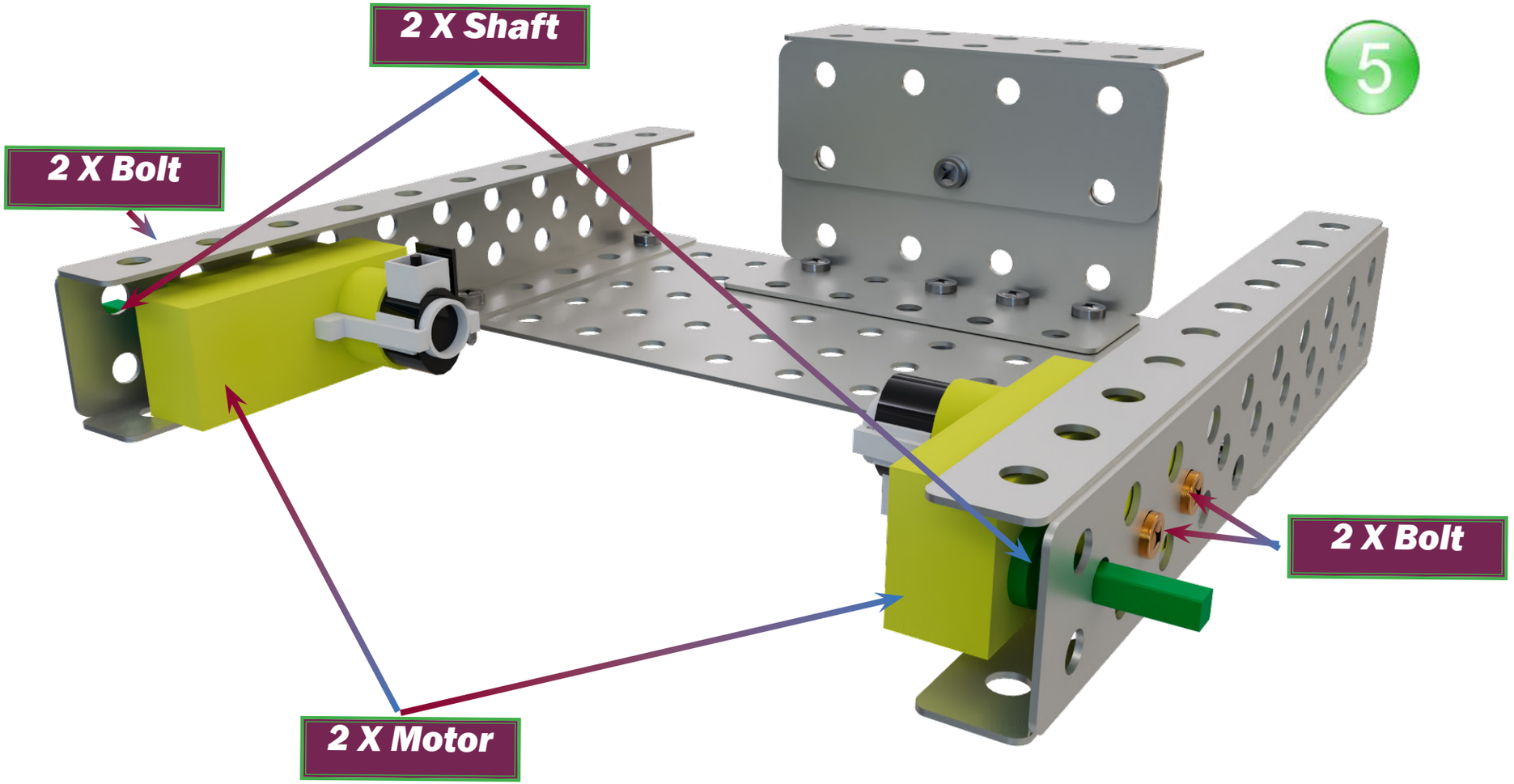


3

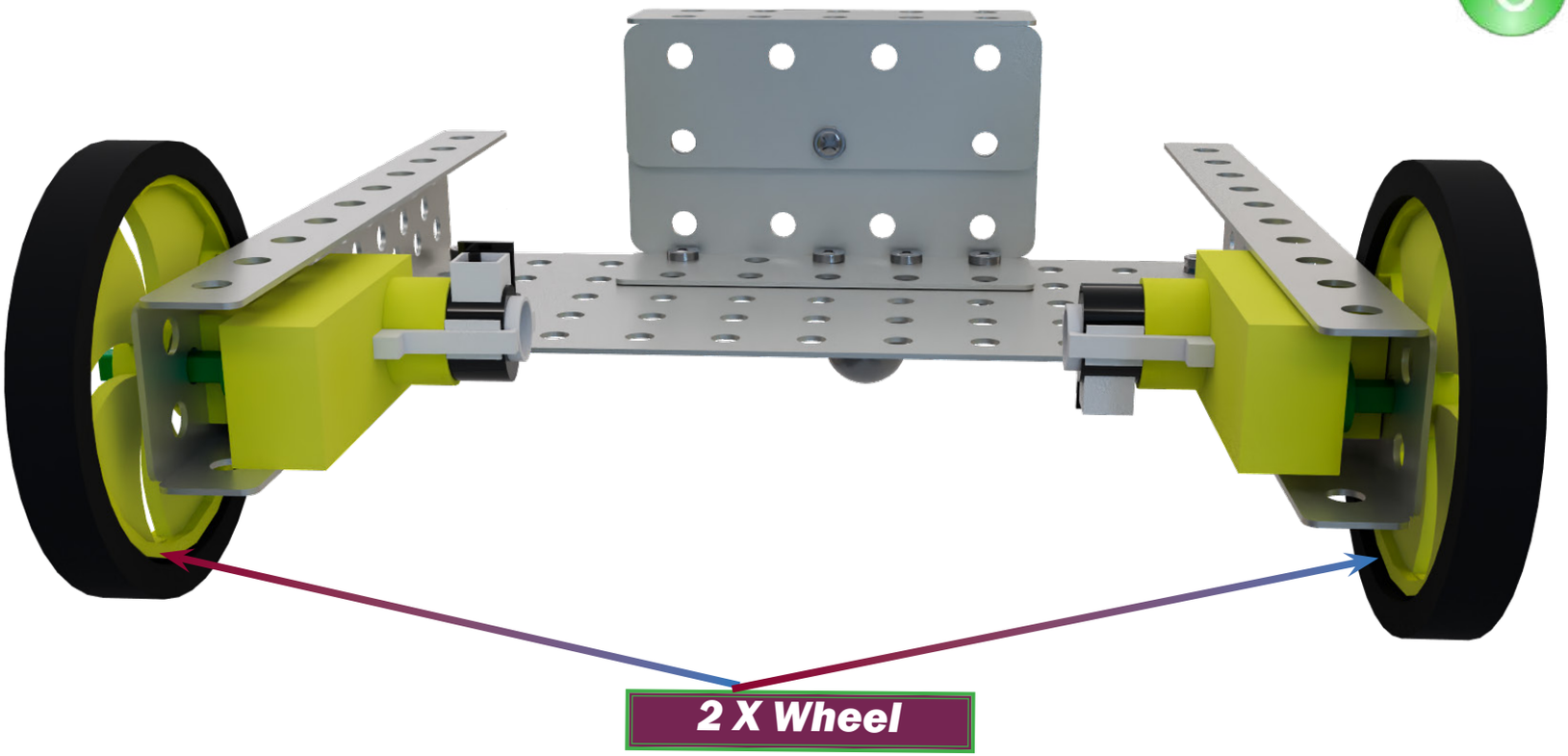


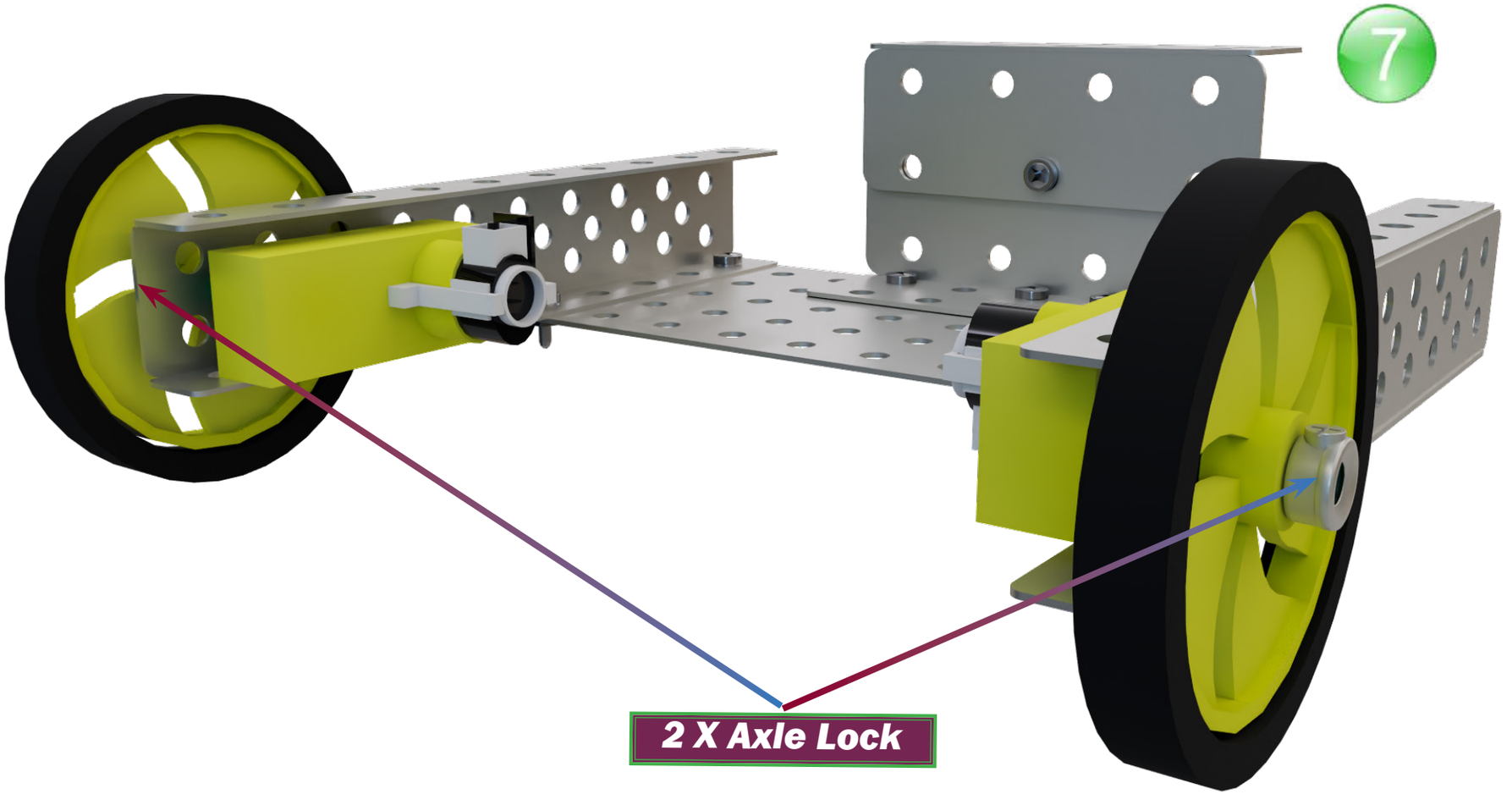


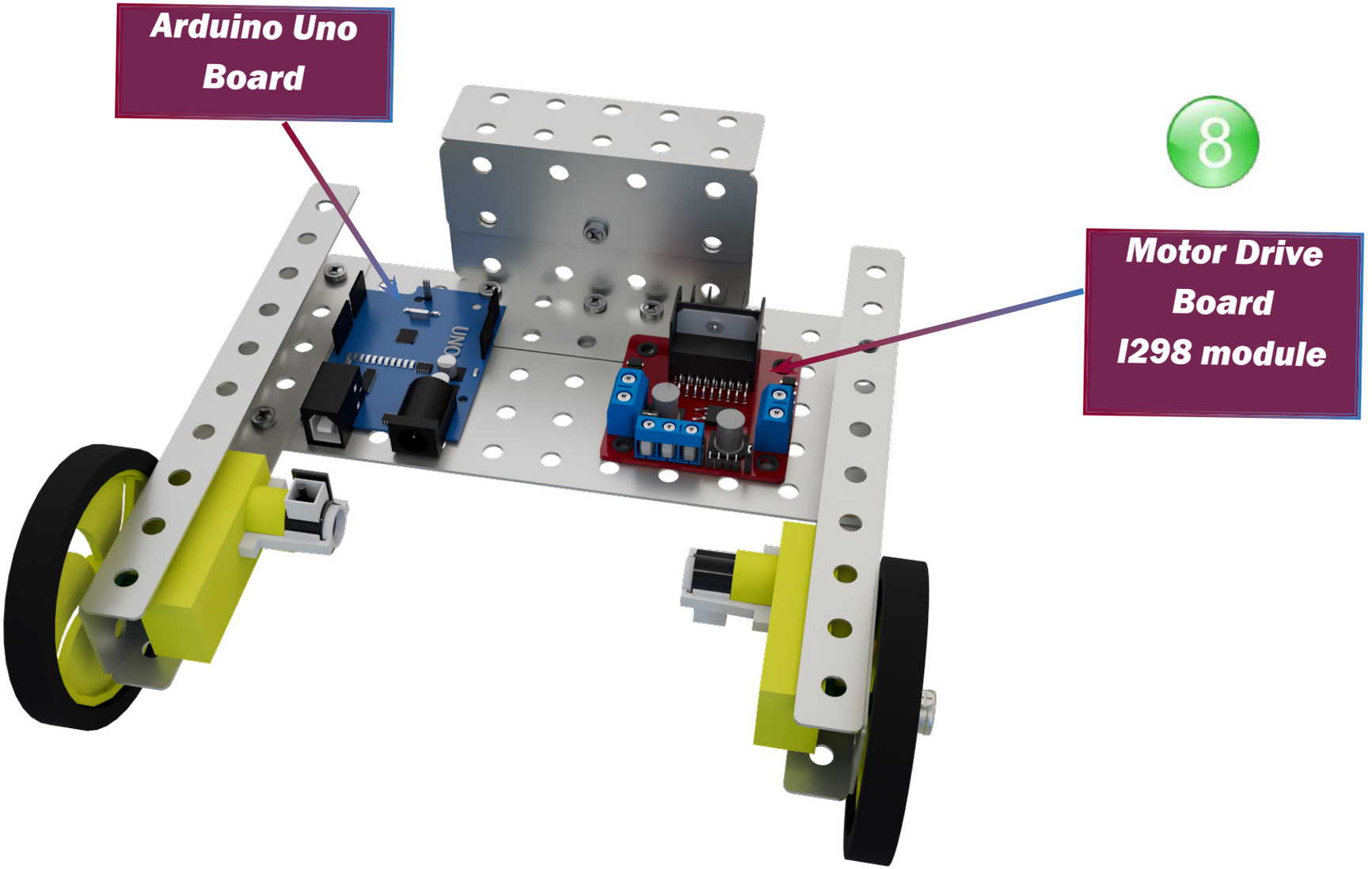
MECHANZ™



6

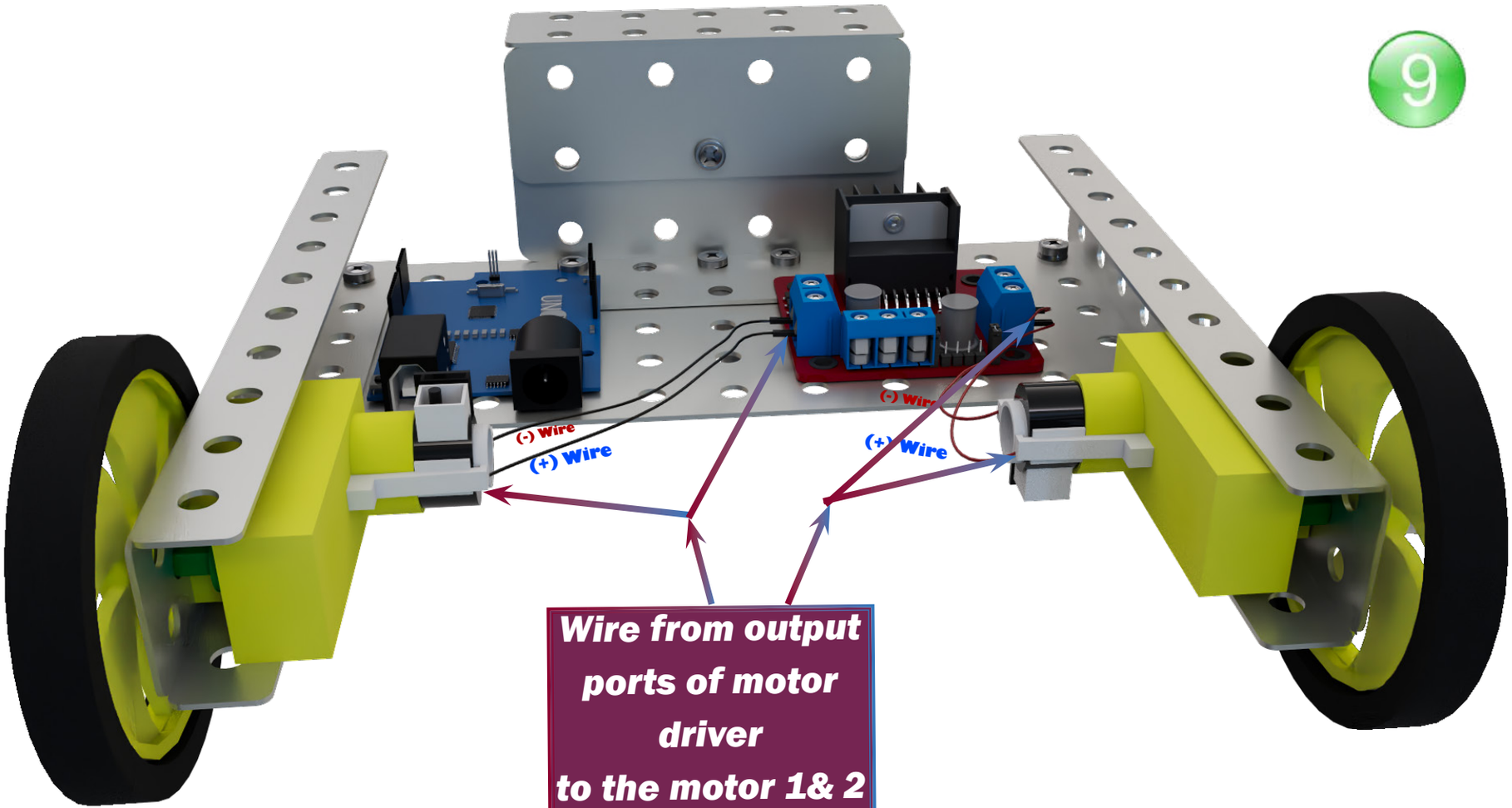




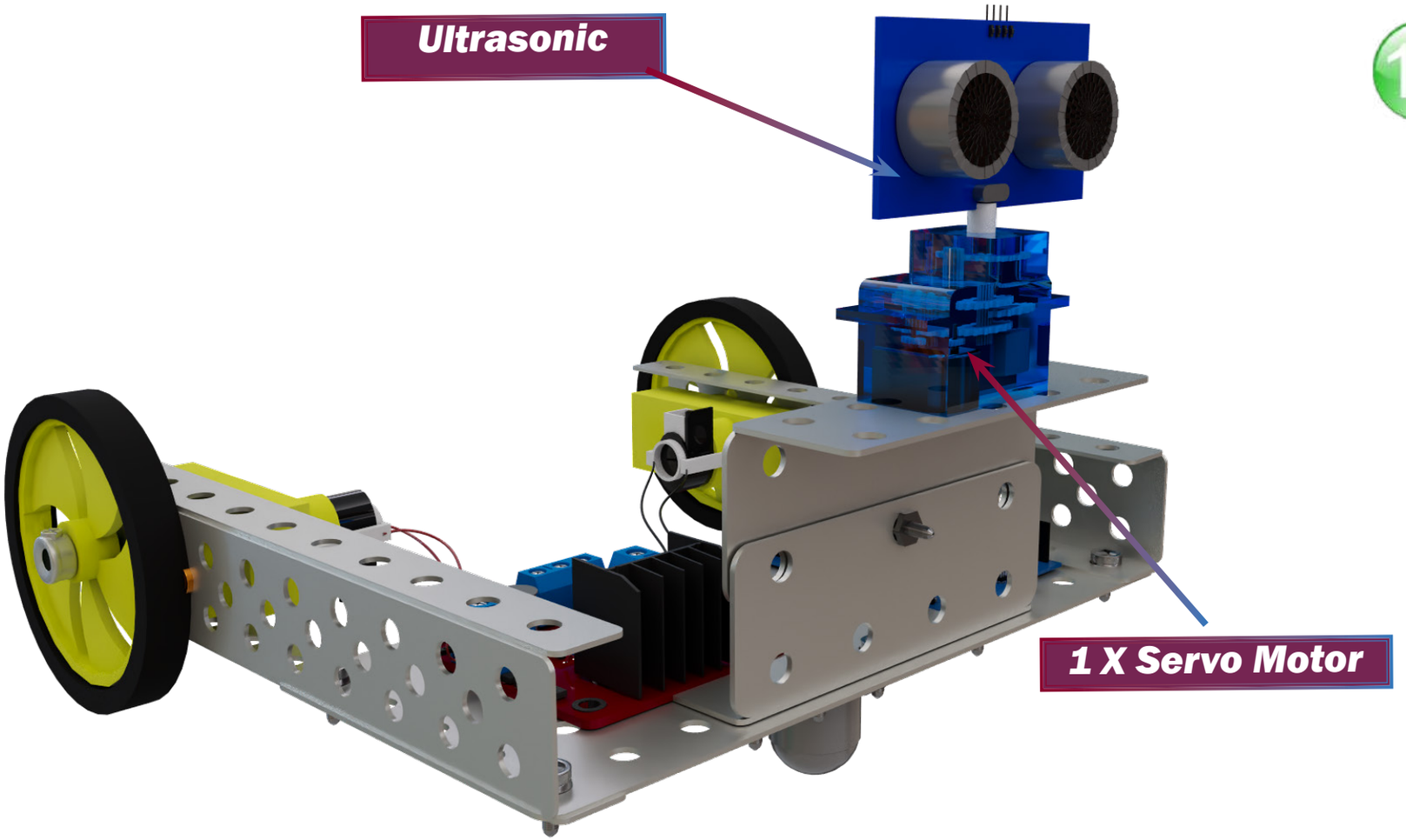


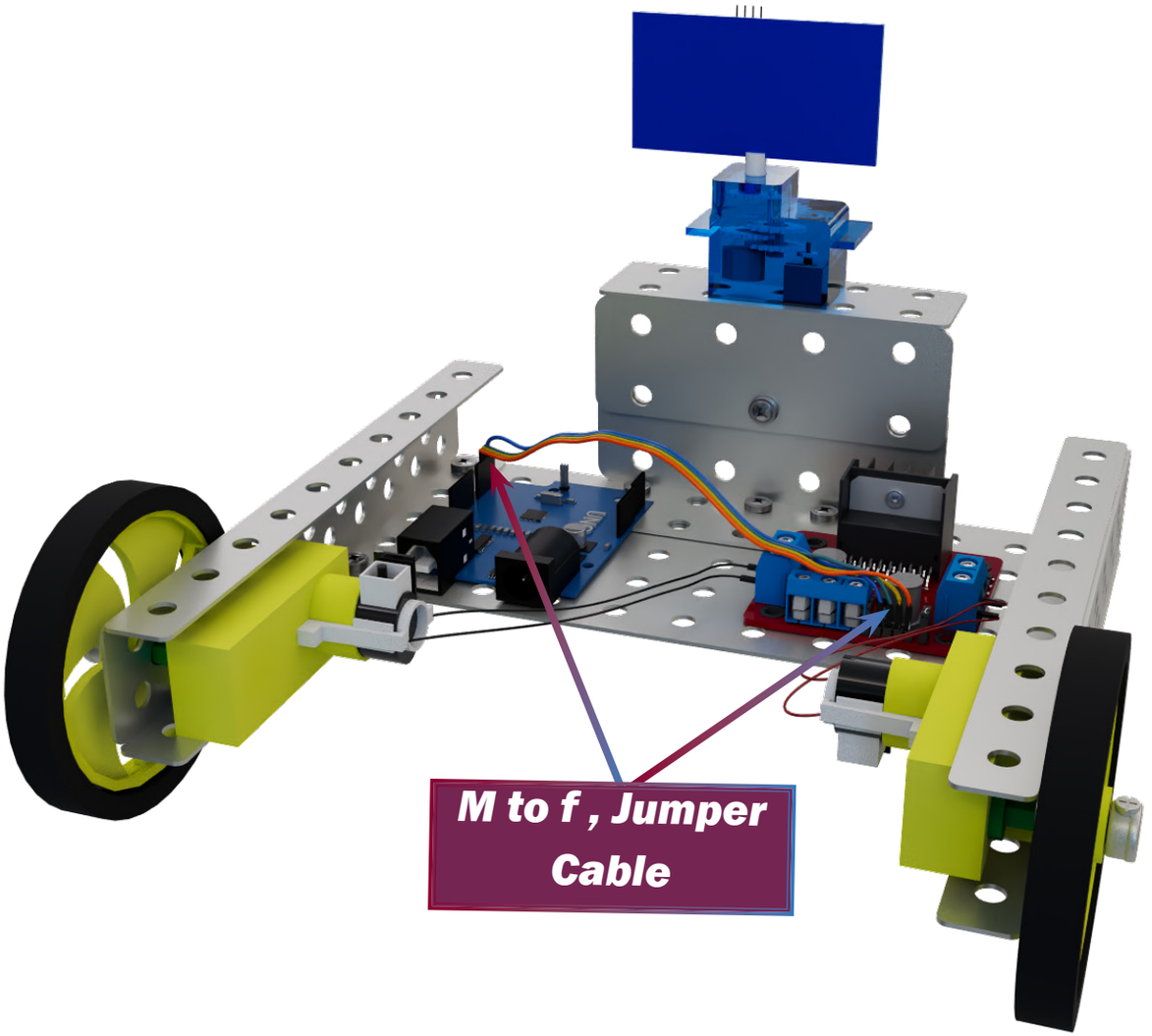
MECHANZ™

9

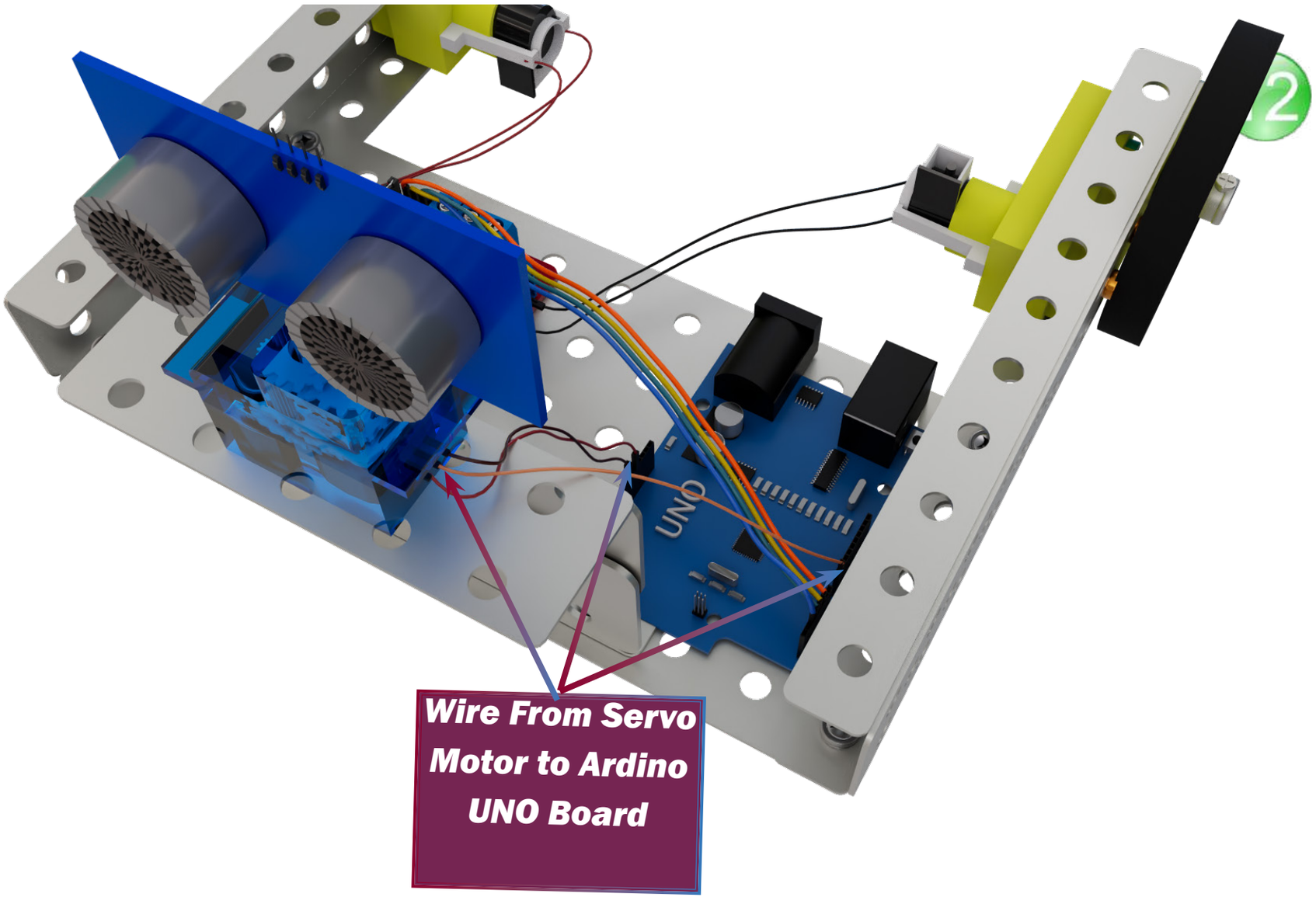


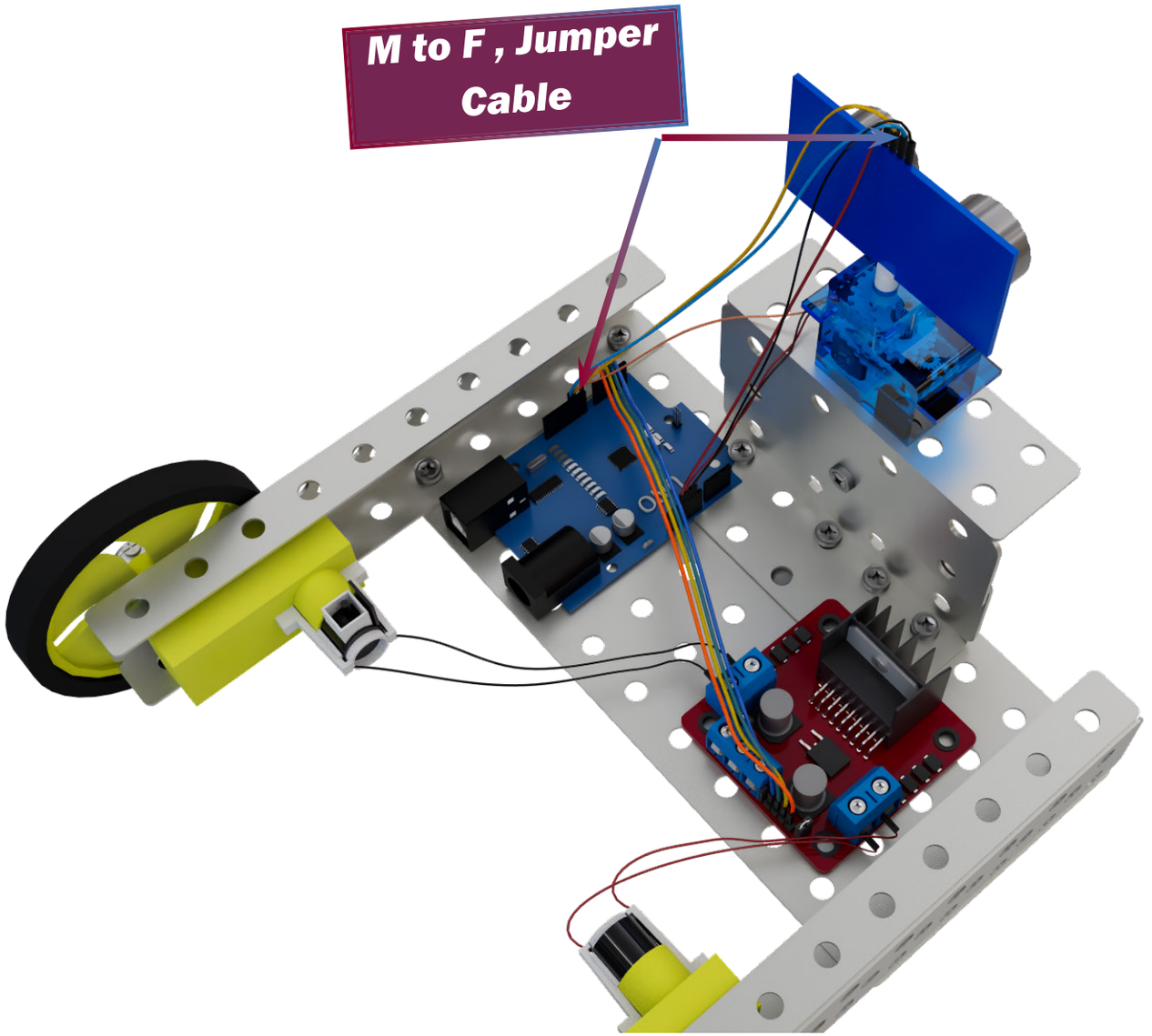
Wire from output ports of motor driver to the motor 1& 2

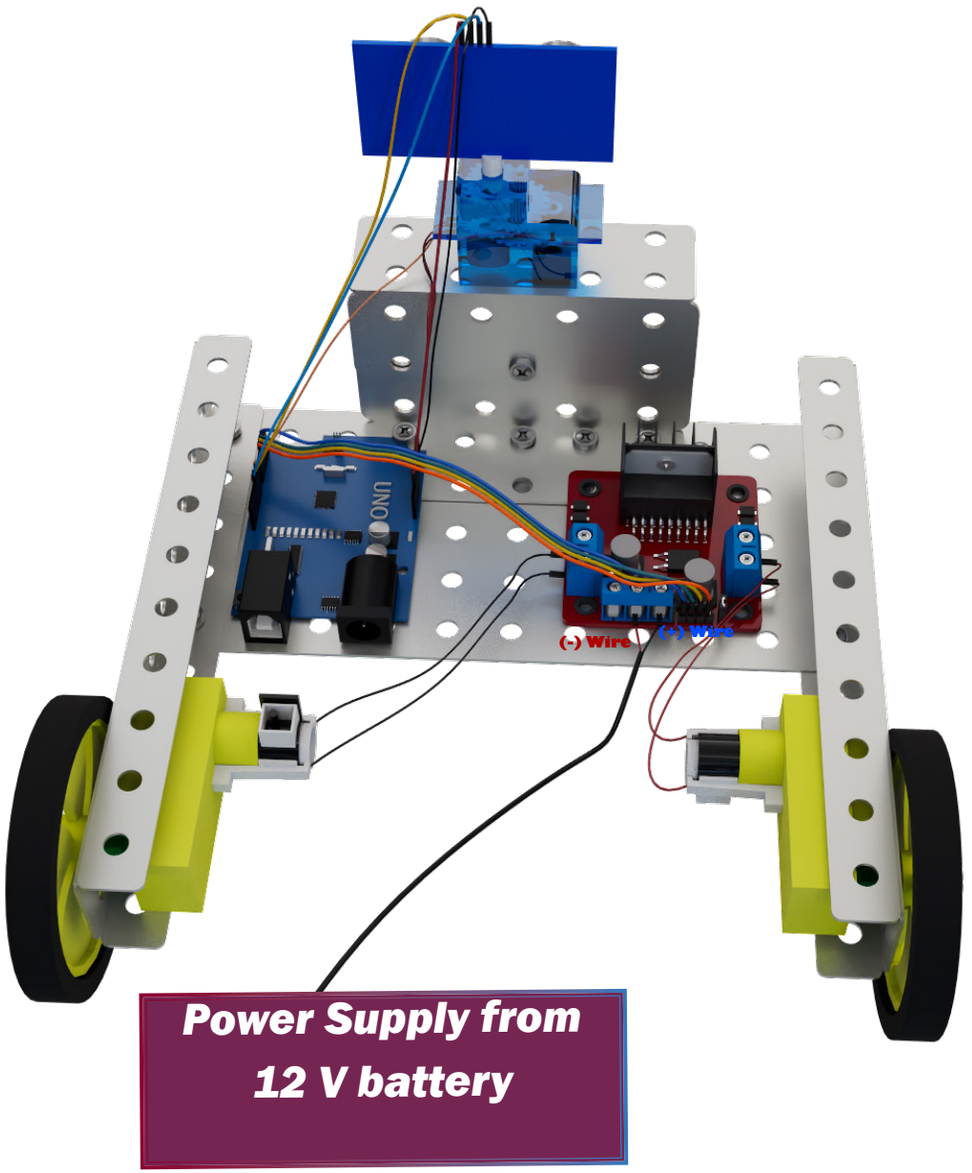




MECHANZ™







Coding For Obstracle Avoiding Bot

```
#include <Servo.h>          //Servo motor library. This is standard library
#include <NewPing.h>        //Ultrasonic sensor function library. You must install this library

//our L298N control pins

const int LeftMotorForward = 7;
const int LeftMotorBackward = 6;
const int RightMotorForward = 4;
const int RightMotorBackward = 5;

//sensor pins
#define trig_pin A1 //analog input 1
#define echo_pin A2 //analog input 2

#define maximum_distance 200
boolean goesForward = false;
int distance = 100;

NewPing sonar(trig_pin, echo_pin, maximum_distance); //sensor function
Servo servo_motor; //our servo name

void setup(){
```

```
  pinMode(RightMotorForward, OUTPUT);
  pinMode(LeftMotorForward, OUTPUT);
  pinMode(LeftMotorBackward, OUTPUT);
  pinMode(RightMotorBackward, OUTPUT);

  servo_motor.attach(10); //our servo pin

  servo_motor.write(115);
  delay(2000);
  distance = readPing();
  delay(100);
  distance = readPing();
  delay(100);
  distance = readPing();
  delay(100);
  distance = readPing();
  delay(100);
}


void loop(){
  int distanceRight = 0;
  int distanceLeft = 0;
  delay(50);

  if (distance <= 20){
    moveStop();
```

```
    delay(300);
    moveBackward();
    delay(400);
    moveStop();
    delay(300);
    distanceRight = lookRight();
    delay(300);
    distanceLeft = lookLeft();
    delay(300);

    if (distance >= distanceLeft){
      turnRight();
      moveStop();
    }
    else{
      turnLeft();
      moveStop();
    }
  }
  else{
    moveForward();
  }
    distance = readPing();
}

int lookRight(){
  servo_motor.write(50);
```

```
  delay(500);
  int distance = readPing();
  delay(100);
  servo_motor.write(115);
  return distance;
}


int lookLeft(){
  servo_motor.write(170);
  delay(500);
  int distance = readPing();
  delay(100);
  servo_motor.write(115);
  return distance;
  delay(100);
}


int readPing(){
  delay(70);
  int cm = sonar.ping_cm();
  if (cm==0){
    cm=250;
  }
  return cm;
}
```

```
void moveStop(){
  digitalWrite(RightMotorForward, LOW);
  digitalWrite(LeftMotorForward, LOW);
  digitalWrite(RightMotorBackward, LOW);
  digitalWrite(LeftMotorBackward, LOW);
}


void moveForward(){
  if(!goesForward){
    goesForward=true;

    digitalWrite(LeftMotorForward, HIGH);
    digitalWrite(RightMotorForward, HIGH);
    digitalWrite(LeftMotorBackward, LOW);
    digitalWrite(RightMotorBackward, LOW);
  }
}


void moveBackward(){
  goesForward=false;

  digitalWrite(LeftMotorBackward, HIGH);
  digitalWrite(RightMotorBackward, HIGH);
  digitalWrite(LeftMotorForward, LOW);
  digitalWrite(RightMotorForward, LOW);
```

```
}

void turnRight(){

  digitalWrite(LeftMotorForward, HIGH);
  digitalWrite(RightMotorBackward, HIGH);
  digitalWrite(LeftMotorBackward, LOW);
  digitalWrite(RightMotorForward, LOW);

  delay(500);

  digitalWrite(LeftMotorForward, HIGH);
  digitalWrite(RightMotorForward, HIGH);
  digitalWrite(LeftMotorBackward, LOW);
  digitalWrite(RightMotorBackward, LOW);
}

void turnLeft(){

  digitalWrite(LeftMotorBackward, HIGH);
  digitalWrite(RightMotorForward, HIGH);
  digitalWrite(LeftMotorForward, LOW);
  digitalWrite(RightMotorBackward, LOW);
```

```
  delay(500);

  digitalWrite(LeftMotorForward, HIGH);
  digitalWrite(RightMotorForward, HIGH);
  digitalWrite(LeftMotorBackward, LOW);
  digitalWrite(RightMotorBackward, LOW);

}
```