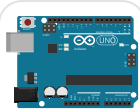


Parking gate: open when a car comes · START A BIT

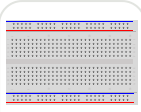


Scan to open

Parts you need



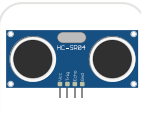
Arduino
UNO



Breadboard



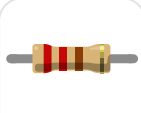
180° servo



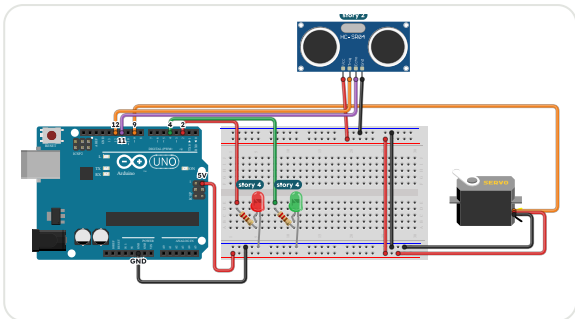
Ultrasonic
sensor (HC-
SR04)



LEDs



220 Ω
resistors



pin 9 → servo signal

5V → + rail → servo

GND → – rails

pin 12 → Trig [story 2](#)

pin 11 → Echo [story 2](#)

5V → rails → VCC [story 2](#)

pin 2 → red LED: [story 4](#)
FULL

pin 4 → green LED: [story 4](#)
free

Remember

Servo

Three wires: brown or black to GND, red to +, orange or yellow is the signal from a pin.

Ultrasonic distance sensor

distance in cm = time in μs $\div$ 2 $\div$ 29.1

State machine: what the program remembers

Keep the state in a variable (like *gateOpen*), change it only when something happens, and act on it every loop.

Story 1: Lift the gate

```
#include <Servo.h>

const int gatePin = 9;           // degrees: the
const int closed = 0;           // degrees: the
Servo gate;

void setup() {
  gate.attach(gatePin);         // tell the Ser
  gate.write(closed);
}

void loop() {
  gate.write(open);             // lift
  delay(2000);
  gate.write(closed);           // lower
  delay(2000);
}
```

Story 2: Open when a car comes

```
#include <HCSR04.h>
#include <Servo.h>

const int triggerPin = 12;
const int echoPin = 11;
const int gatePin = 9;
const int closed = 0;
const int open = 90;
const int carIsNear = 15;       // cm
UltraSonicDistanceSensor sensor(triggerPin, echoPin);
Servo gate;

void setup() {
  Serial.begin(9600);
  gate.attach(gatePin);
  gate.write(closed);
}

void loop() {
  float distance = sensor.measureDistanceCm();
  Serial.println(distance);
  if (distance > 0 && distance < carIsNear) { //
    gate.write(open);
  } else {
    gate.write(closed);
  }
  delay(100);
}
```

Story 3: Close after the car has passed

```
#include <HCSR04.h>
#include <Servo.h>

const int triggerPin = 12;
const int echoPin = 11;
const int gatePin = 9;
const int closed = 0;
const int open = 90;
const int carIsNear = 15;       // cm: open
const int carIsGone = 30;       // cm: further
const int closeAfter = 30;      // loops of 100
UltraSonicDistanceSensor sensor(triggerPin, echoPin);
Servo gate;
bool gateOpen = false;         // the state: w
int clearFor = 0;              // how many loc

void setup() {
  gate.attach(gatePin);
  gate.write(closed);
}

void loop() {
  float distance = sensor.measureDistanceCm();
  bool near = distance > 0 && distance < carIsNear;
  bool gone = distance < 0 || distance > carIsGone;
  if (near) {                   // a car: open,
    gateOpen = true;
    clearFor = 0;
    gate.write(open);
  } else if (gateOpen && gone) { // open, and no
    clearFor = clearFor + 1;
    if (clearFor >= closeAfter) {
      gateOpen = false;
      gate.write(closed);
    }
  }
  delay(100);
}
```

Story 4: Count the cars, show FULL

```
#include <HCSR04.h>
#include <Servo.h>

const int triggerPin = 12;
const int echoPin = 11;
const int gatePin = 9;
const int redLedPin = 2;         // FULL
const int greenLedPin = 4;       // places free
const int closed = 0;
const int open = 90;
const int carIsNear = 15;
const int carIsGone = 30;
const int closeAfter = 30;
const int places = 3;           // how many car
UltraSonicDistanceSensor sensor(triggerPin, echoPin);
Servo gate;
bool gateOpen = false;
bool carWasNear = false;       // the state of
int clearFor = 0;
int cars = 0;                  // how many car

void setup() {
  Serial.begin(9600);
  pinMode(redLedPin, OUTPUT);
  pinMode(greenLedPin, OUTPUT);
  gate.attach(gatePin);
  gate.write(closed);
}

void loop() {
  float distance = sensor.measureDistanceCm();
  bool near = distance > 0 && distance < carIsNear;
  bool gone = distance < 0 || distance > carIsGone;
  if (carWasNear && !near && gateOpen) { // it was
    cars = cars + 1;
    Serial.print("cars inside: ");
    Serial.println(cars);
  }
  bool full = cars >= places;
  digitalWrite(redLedPin, full);
  digitalWrite(greenLedPin, !full);
  if (near && !full) {
    gateOpen = true;
    clearFor = 0;
    gate.write(open);
  } else if (gateOpen && full) { // the la
    gateOpen = false;
    gate.write(closed);
  } else if (gateOpen && gone) {
    clearFor = clearFor + 1;
    if (clearFor >= closeAfter) {
      gateOpen = false;
      gate.write(closed);
    }
  }
  carWasNear = near;           // rememb
  delay(100);
}
```