**Universidad politÉcnica de Tulancingo**

**INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES**

**CONTROL DE PUERTAS AUTOMATICAS**

**ALINCORL**

**ENRIQUE DIEGO CASTRO**

**Y**

**BRIAN ALEXIS ESCAMILLA TORRES**

**MAYO - AGOSTO 2020**



**U**

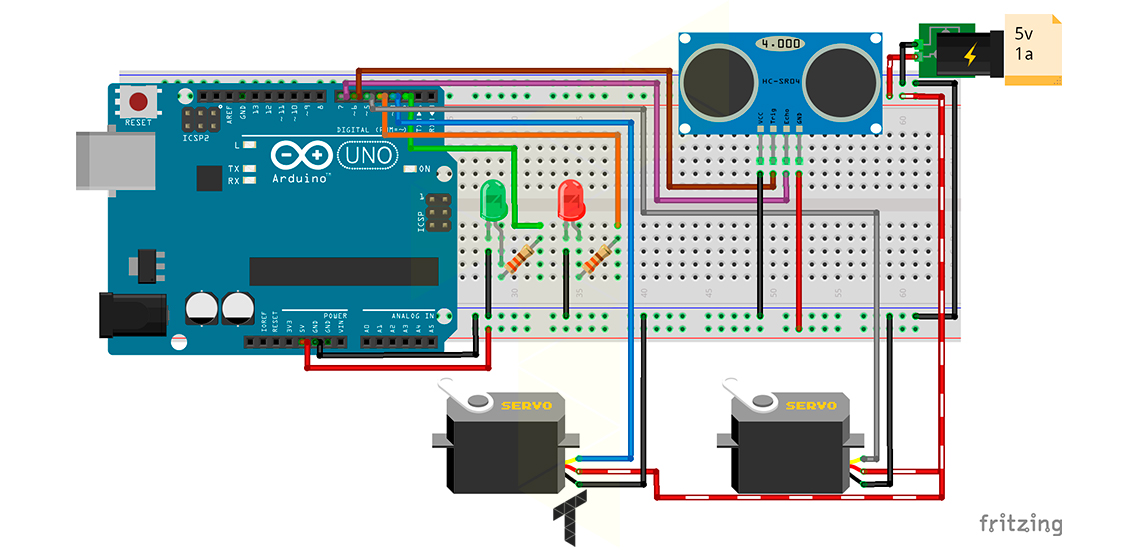
**P**

**T**



**Diagrama**

El siguiente esquemático muestra cómo se debe conectar todos los componentes con la placa



### Código

 Este es el código que se ocupara para poder programar la puerta esta comentado de tal forma que puedas entender su estructura y función.

//The TRIG and ECHO pins of the ultrasonic are defined and

//servoRightT & servoLeftT objects are created to control the servos

Ultrasonic ultrasonicT(6, 7);

Servo servoRightT;

Servo servoLeftT;

//Assigning the pins to the LEDs

const uint8\_t ledOpenT = 2;

const uint8\_t ledClosedT = 4;

//Variables to store positions of the servos, the distance of the ultrasonic, a flag that

//allows to rectify the change of distance and starts a variable of minimum seconds to two

int16\_t posRightT, posLeftT, distanceT, initialDistanceT, continuousSecondsT = 0;

bool flagT = false;

const uint8\_t minimalSecondsT = 2;

//loopTimeT and waitingDoorClosingT define the time (in milliseconds) that the events of

//your function will last, timeElapsedT and timeElapsedDoorClosingT are variables

//that will store the elapsed time

const uint8\_t loopTimeT = 200;

unsigned long timeElapsedT = 0;

const uint16\_t waitingDoorClosingT = 1000;

unsigned long timeElapsedDoorClosingT = 0;

void setup() {

//The pins of the servos are defined

servoLeftT.attach(3);

servoRightT.attach(5);

//The servos, initially, will move 90 degrees

servoLeftT.write(90);

servoRightT.write(90);

pinMode(ledOpenT, OUTPUT);

pinMode(ledClosedT, OUTPUT);

digitalWrite(ledClosedT, HIGH);

//The variables invoke the millis() action

timeElapsedT = millis();

timeElapsedDoorClosingT = millis();

}

void loop() {

//A variable that calls the milli() function is created, then the function overflow is managed

unsigned long currentMillisLoopT = millis();

if ((unsigned long)(currentMillisLoopT - timeElapsedT) >= loopTimeT) {

//The distance is obtained in real time and stored in distanceT to be compared with

//the return value sumary(), if both distances are equal and the flag is true then

//the door is opened, otherwise, call the function beforeCloseDoor()

distanceT = ultrasonicT.Ranging(CM);

if (distanceT == summary()) {

if (distanceT < 10 && flagT == false)

openDoor();

else if (distanceT >= 10 && flagT == true)

beforeCloseDoor();

}

timeElapsedT = millis();

}

}

//This function returns the average of 4 readings of the distance, its purpose is to have

//a more accurate data of the measurement

uint8\_t summary() {

uint8\_t sumT = 0;

for (uint8\_t iT = 0; iT < 3; iT ++) {

sumT = sumT + (distanceT = ultrasonicT.Ranging(CM));

delay(50);

}

initialDistanceT = sumT / 3;

return (initialDistanceT);

}

//openDoor() generates 2 events, one is in the servomotors to change its position

//(both in opposite way) and another event changes the state of the LEDs

void openDoor() {

flagT = true;

posLeftT = 90;

for (posRightT = 90; posRightT >= 0; posRightT -= 1) {

if (posLeftT <= 180) {

posLeftT++;

servoLeftT.write(posLeftT);

}

servoRightT.write(posRightT);

delay(15);

}

digitalWrite(ledOpenT, HIGH);

digitalWrite(ledClosedT, LOW);

}

//A timeout of +-3 seconds is granted and calls the closeDoor() function

//You can modify the wait time in the variable minimalSecondsT

void beforeCloseDoor() {

unsigned long currentMillisDoorT = millis();

if ((unsigned long)(currentMillisDoorT - timeElapsedDoorClosingT) >= waitingDoorClosingT ) {

continuousSecondsT++;

if (continuousSecondsT == minimalSecondsT)

closeDoor();

timeElapsedDoorClosingT = millis();

}

}

//This works in a manner contrary to the openDoor() function

void closeDoor() {

flagT = false;

posLeftT = 180;

for (posRightT = 0; posRightT <= 90; posRightT += 1) {

if (posLeftT >= 90) {

posLeftT--;

servoLeftT.write(posLeftT);

}

servoRightT.write(posRightT);

delay(15);

}

continuousSecondsT = 0;

digitalWrite(ledClosedT, HIGH);

digitalWrite(ledOpenT, LOW);

}