Anti-theft system for vending machine

**Author: Nguyễn Cao Huy Hoàng – HE153231**

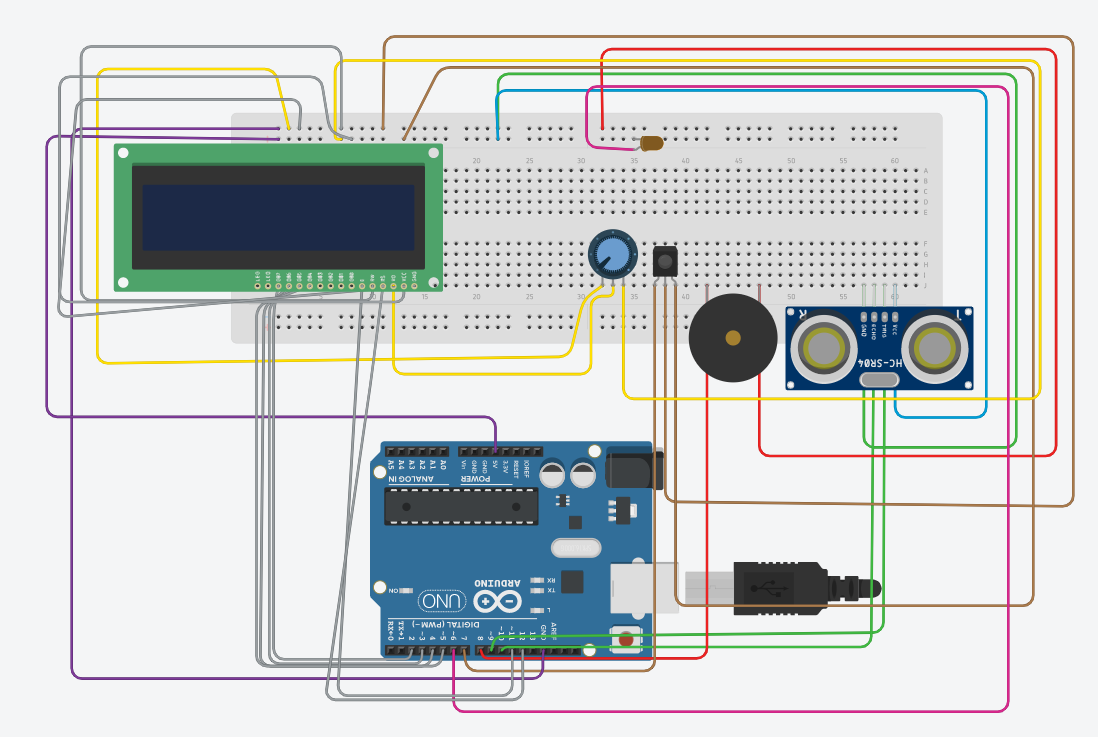
**Nguyễn Tấn Việt – HE153763**

Anxious with naughty and reckless thieves making your vending machine empty in the blink of an eye by just a hammer? Let this anti-theft system stop them for you! This system’s working depends on an ultrasonic range finder called SRF05. It can detect the distance of the closest object in front of the sensor from 3 cm up to 400 cm, but for the purpose of anti-theft, we restrict the range down to 5 cm so that the ultrasonic can trigger the alerting LED to shine and the alarm whistle to make some noises. Simultaneously, the LCD display will change its status from allowing inputting to displaying the warning of distance between the locker and customer, then they will know how far they need to recede in order not to trigger the alerting LED and alarm whistle. The LCD display also restrict inputted number down to three, which prevent invalid inputs.

Hardware Required

* Arduino Board.
* Ultrasonic Range Finder SRF05.
* Hook-up wires.
* LED.
* Potentionmeter B5K.
* LCD display.
* IR Remote.
* IR Remote Control.
* Whistle Detector.

Circuit



Code

// include the library code:

#include <LiquidCrystal.h>

#include <IRremote.h>

#include <stdbool.h>

// initialize the library by associating any needed LCD interface pin

// with the arduino pin number it is connected to

const int RECV\_PIN = 7;

float sinVal;

int toneVal;

IRrecv irrecv(RECV\_PIN);

decode\_results results;

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

unsigned long key\_value = 0;

const unsigned int TRIG\_PIN = 9;

const unsigned int ECHO\_PIN = 10;

const unsigned int BAUD\_RATE = 9600;

#define NOTE\_B0 31

#define NOTE\_C1 33

#define NOTE\_CS1 35

#define NOTE\_D1 37

#define NOTE\_DS1 39

#define NOTE\_E1 41

#define NOTE\_F1 44

#define NOTE\_FS1 46

#define NOTE\_G1 49

#define NOTE\_GS1 52

#define NOTE\_A1 55

#define NOTE\_AS1 58

#define NOTE\_B1 62

#define NOTE\_C2 65

#define NOTE\_CS2 69

#define NOTE\_D2 73

#define NOTE\_DS2 78

#define NOTE\_E2 82

#define NOTE\_F2 87

#define NOTE\_FS2 93

#define NOTE\_G2 98

#define NOTE\_GS2 104

#define NOTE\_A2 110

#define NOTE\_AS2 117

#define NOTE\_B2 123

#define NOTE\_C3 131

#define NOTE\_CS3 139

#define NOTE\_D3 147

#define NOTE\_DS3 156

#define NOTE\_E3 165

#define NOTE\_F3 175

#define NOTE\_FS3 185

#define NOTE\_G3 196

#define NOTE\_GS3 208

#define NOTE\_A3 220

#define NOTE\_AS3 233

#define NOTE\_B3 247

#define NOTE\_C4 262

#define NOTE\_CS4 277

#define NOTE\_D4 294

#define NOTE\_DS4 311

#define NOTE\_E4 330

#define NOTE\_F4 349

#define NOTE\_FS4 370

#define NOTE\_G4 392

#define NOTE\_GS4 415

#define NOTE\_A4 440

#define NOTE\_AS4 466

#define NOTE\_B4 494

#define NOTE\_C5 523

#define NOTE\_CS5 554

#define NOTE\_D5 587

#define NOTE\_DS5 622

#define NOTE\_E5 659

#define NOTE\_F5 698

#define NOTE\_FS5 740

#define NOTE\_G5 784

#define NOTE\_GS5 831

#define NOTE\_A5 880

#define NOTE\_AS5 932

#define NOTE\_B5 988

#define NOTE\_C6 1047

#define NOTE\_CS6 1109

#define NOTE\_D6 1175

#define NOTE\_DS6 1245

#define NOTE\_E6 1319

#define NOTE\_F6 1397

#define NOTE\_FS6 1480

#define NOTE\_G6 1568

#define NOTE\_GS6 1661

#define NOTE\_A6 1760

#define NOTE\_AS6 1865

#define NOTE\_B6 1976

#define NOTE\_C7 2093

#define NOTE\_CS7 2217

#define NOTE\_D7 2349

#define NOTE\_DS7 2489

#define NOTE\_E7 2637

#define NOTE\_F7 2794

#define NOTE\_FS7 2960

#define NOTE\_G7 3136

#define NOTE\_GS7 3322

#define NOTE\_A7 3520

#define NOTE\_AS7 3729

#define NOTE\_B7 3951

#define NOTE\_C8 4186

#define NOTE\_CS8 4435

#define NOTE\_D8 4699

#define NOTE\_DS8 4978

void setup() {

// set up the LCD's number of columns and rows:

lcd.begin(16, 2);

Serial.begin(9600);

irrecv.enableIRIn();

irrecv.blink13(true);

pinMode(TRIG\_PIN, OUTPUT);

pinMode(ECHO\_PIN, INPUT);

pinMode(8, OUTPUT);

pinMode(6, OUTPUT);

lcd.cursor();

}

void print\_input\_digit(int value, int count)

{

lcd.setCursor(count - 1, 0);

lcd.print(value);

}

int input\_number(String command)

{

int num[2] = {0, 0};

int count, number, value;

while(true)

{

alert();

count = 11, number = 0, value = -1;

lcd.clear();

lcd.setCursor(0, 0);

lcd.print(command);

//Input number

while(true)

{

if(alert()){

lcd.setCursor(0, 0);

lcd.print(command);

num[0]=0;

num[1]=0;

count=11;

number=0;

};

if(irrecv.decode(&results))

{

Serial.println(results.value, HEX);

if(results.value == 0xFF6897) //0

{

value = 0;

++count;

if(count-11 <= 2)

{

print\_input\_digit(value, count);

num[count-11 - 1] = value;

}

}

else if(results.value == 0xFF30CF) //1

{

value = 1;

++count;

if(count-11 <= 2)

{

print\_input\_digit(value, count);

num[count -11- 1] = value;

}

}

else if(results.value == 0xFF18E7) //2

{

value = 2;

++count;

if(count-11 <= 2)

{

print\_input\_digit(value, count);

num[count -11- 1] = value;

}

}

else if(results.value == 0xFF7A85) //3

{

value = 3;

++count;

if(count-11 <= 2)

{

print\_input\_digit(value, count);

num[count -11- 1] = value;

}

}

else if(results.value == 0xFF10EF) //4

{

value = 4;

++count;

if(count-11 <= 2)

{

print\_input\_digit(value, count);

num[count -11- 1] = value;

}

}

else if(results.value == 0xFF38C7) //5

{

value = 5;

++count;

if(count-11 <= 2)

{

print\_input\_digit(value, count);

num[count -11- 1] = value;

}

}

else if(results.value == 0xFF5AA5) //6

{

value = 6;

++count;

if(count-11 <= 2)

{

print\_input\_digit(value, count);

num[count -11- 1] = value;

}

}

else if(results.value == 0xFF42BD) //7

{

value = 7;

++count;

if(count-11 <= 2)

{

print\_input\_digit(value, count);

num[count -11- 1] = value;

}

}

else if(results.value == 0xFF4AB5) //8

{

value = 8;

++count;

if(count -11<= 2)

{

print\_input\_digit(value, count);

num[count -11- 1] = value;

}

}

else if(results.value == 0xFF52AD) //9

{

value = 9;

++count;

if(count-11 <= 2)

{

print\_input\_digit(value, count);

num[count -11- 1] = value;

}

}

else if(results.value == 0xFFA857) //enter

{

irrecv.resume();

lcd.clear();

break;

}

irrecv.resume();

}

}

if(count-11 == 0)

{

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("No code");

delay(1000);

continue;

}

else if(count-11 >= 2)

{

number = num[0] \* 10 + num[1];

}

// Serial.println(number);

//Validate the input number

break;

}

return number;

}

bool alert() {

digitalWrite(TRIG\_PIN, LOW);

delayMicroseconds(2);

digitalWrite(TRIG\_PIN, HIGH);

delayMicroseconds(10);

digitalWrite(TRIG\_PIN, LOW);

unsigned long duration = pulseIn(ECHO\_PIN, HIGH);

int distance = duration / 29 / 2;

if (duration == 0) {

lcd.println("no pulse");

lcd.clear();

return false;

}

else if (distance <= 5) {

digitalWrite(6, HIGH);

int melody[] = {

NOTE\_C4, NOTE\_G3, NOTE\_G3, NOTE\_A3, NOTE\_G3, 0, NOTE\_B3, NOTE\_C4

};

// note durations: 4 = quarter note, 8 = eighth note, etc.:

int noteDurations[] = {

4, 8, 8, 4, 4, 4, 4, 4

};

for (int thisNote = 0; thisNote < 8; thisNote++) {

int noteDuration = 1000 / noteDurations[thisNote];

tone(8, melody[thisNote], noteDuration);

int pauseBetweenNotes = noteDuration \* 1.30;

delay(pauseBetweenNotes);

// stop the tone playing:

noTone(8);

lcd.clear();

lcd.print("STAY AWAY!");

}

delay(3000);

lcd.clear();

digitalWrite(6, LOW);

return true;

}

}

void loop() {

// set the display to automatically scroll:

lcd.noAutoscroll();

alert();

int Min=0,Max=50;

int code = input\_number("Enter code:");

if(code < Min || code > Max)

{

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Wrong Code!");

delay(2000);

}

else{

lcd.print("PLACE ");

lcd.print(code);

lcd.print(" SUCCESS");

delay(1000);

}

alert();

irrecv.resume();

delay(100);

}