Introduction to Internet of Things (IoT) 243-2E5-DW

Instructor:

Carlos Mendes

**PBL: Door Alarm System**

Axel Tille-Ascencio – 2443059

Corrado Palermo – 2445543

Submission date:

May 13, 2025

**Project Description:**

This project is a smart door alarm system. It uses an ultrasonic sensor to detect whether a door is open, and the buzzer will notify the user. If the door stays open for a certain amount of time, a buzzer will go off. The alarm turns off when you close the door.

**Problem it solves:**

Sometimes people forget to close the door properly. This can be unsafe, especially for the front door, as it could lead to intruders or pets escaping. For example, kids might not shut the door all the way, or you might think the door is closed when your hands are full of groceries or bags. These small moments can lead to big safety problems. Our system helps keep the house secure by warning you with a sound if the door is left open for too long. It acts as a simple reminder to double-check the door, adding an extra layer of security to your home.

**Inputs:**

Ultrasonic sensor: Measures the distance between the door and the sensor to check if the door is open or closed.

Button Switch Module: Lets the user turn off the buzzer by pressing it.

**Outputs:**

Active Buzzer: Makes a sound to warn you if the door is left open too long.

**What each function does:**

1. isDoorOpen(int distance)

This checks if the door is open. Measures the distance between the ultrasonic sensor and the door. If the distance is more than 3 cm, it will say that the door is open.

1. buzzcause(bool shouldBuzz)

This function turns the buzzer on or off. If shouldBuzz is true, that means the buzzer will turn on. If it’s false, the buzzer turns off.

**How the sensors and outputs work:**

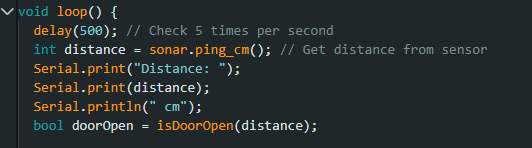
Ultrasonic sensor

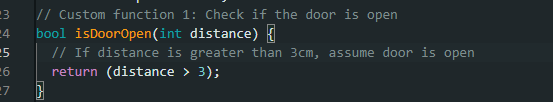
This input sensor sends out a sound wave at a high frequency. The transducer part of the sensor receives and sends ultrasonic sound. The sensor finds the distance between itself and its target using time lapses between the ultrasonic pulses.

Active Buzzer

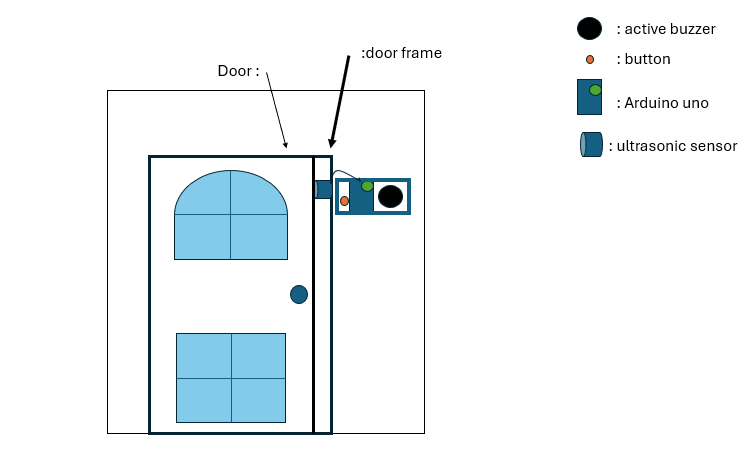
This output buzzer works by making it HIGH when not pressed, or LOW when pressed. This active buzzer has an internal pullup resistor, so the code would also need to include INPUT\_PULLUP for it to work.

**Code structure:**

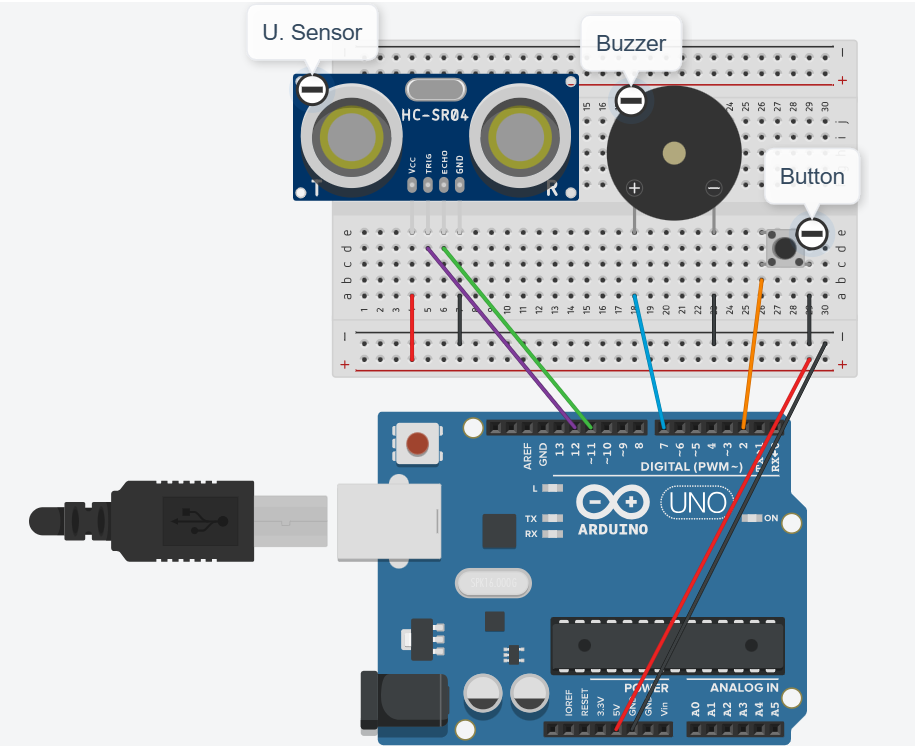
The code structure is based around the distance captured by the ultrasonic sensor, what is collected then goes through custom function 1 which check if the distance is less than or more than 5cm if it's more the door is considered open, and the buzzer will activate within the next 30 seconds. 



**Assembly diagram:**



**Circuit diagram:**



**Things we learned for this code:**

State change detection: Used to detect when a digital input changes from one state to another. It helps your program do something only when the change happens, not the whole time it's the same. This way, the program doesn't keep repeating the same action over and over.

Unsigned long: These are variables that are extended for number storage. They store up to 32 bits and only store positive numbers. For this code, we thought we could use just long because it can store large whole numbers.

**Security:**

This project was designed to improve home security by alerting the user if a door is left open for too long. It can help prevent break-ins, accidents, or letting pets out by mistake. The system is meant to be a simple way to keep your home safer using basic sensors and alerts.

**Disclaimer:**  
 This device is a basic security aid and should not be relied on as the only protection for your home. It does not replace proper locks, alarms, or human supervision.

**References:**

* IoT lab-6
* IoT lab-5
* 37 SENSOR KIT TUTORIAL FOR UNO AND MEGA v2.0.0.19.09.17.pdf
* <https://maxbotix.com/blogs/blog/how-ultrasonic-sensors-work?srsltid=AfmBOopLqXEOMjzlXr3RetLk4df9G9q7r_MvDCEKChLLHksYbnOsqNp7>
* <https://www.circuitbasics.com/how-to-use-active-and-passive-buzzers-on-the-arduino/#:~:text=Explanation%20of%20the%20Code,equal%20to%20Arduino%20pin%208>.
* <https://docs.arduino.cc/built-in-examples/digital/StateChangeDetection/>
* <https://docs.arduino.cc/language-reference/en/variables/data-types/unsignedLong/>

// Door Alarm System

// Introduction to IoT - PBL

// Authors: Axel Tille-Ascencio, Corrado Palermo

// Date: May 6, 2025

//FULL ONE

#include <NewPing.h> // External library for ultrasonic sensor

//Names and set values for parts of the circuit

#define TRIGGER\_PIN 12

#define ECHO\_PIN 11

#define MAX\_DISTANCE 200

#define BUZZER\_PIN 7

#define BUTTON\_PIN 2

// Time (in milliseconds) the door must be open before buzzer sounds

#define DOOR\_OPEN\_DELAY 10000 // 10 seconds

//Creates a sonar object using the NewPing library

NewPing sonar(TRIGGER\_PIN, ECHO\_PIN, MAX\_DISTANCE);

// Functions to track door state and timing

bool buzzerSilenced = false; //Goes true if button is pressed to silence buzzer

long doorOpenTime = 0; //big number type used to measure time in milliseconds

bool doorWasOpen = false;

// Function 1: Check if the door is open

bool isDoorOpen(int distance) { //This function takes a number and returns true if > 3

return (distance > 3); // If distance is greater than 3cm, assume door is open

}

// Function 2: Control the buzzer

void controlBuzzer(bool shouldBuzz) {

if (shouldBuzz) {

digitalWrite(BUZZER\_PIN, HIGH); // Turn buzzer on

} else {

digitalWrite(BUZZER\_PIN, LOW); // Turn buzzer off

}

}

void setup() {

pinMode(BUZZER\_PIN, OUTPUT);

pinMode(TRIGGER\_PIN, OUTPUT);

pinMode(ECHO\_PIN, INPUT);

pinMode(BUTTON\_PIN, INPUT\_PULLUP); // Use internal pull-up resistor

Serial.begin(9600);

}

void loop() {

delay(200); // Slows down how often system checks the door

int distance = sonar.ping\_cm(); // Get distance from sensor

Serial.print("Distance: ");

Serial.print(distance);

Serial.println(" cm");

bool doorOpen = isDoorOpen(distance); // Calls said function with the distance that was measured

// Check if button is pressed (LOW means pressed)

if (digitalRead(BUTTON\_PIN) == LOW) {

buzzerSilenced = true;

controlBuzzer(false); // Turn off buzzer

Serial.println("Buzzer silenced by button.");

}

// If door is closed, reset everything

if (!doorOpen) {

doorOpenTime = 0;

buzzerSilenced = false;

controlBuzzer(false); // Turn off buzzer, resets timer

}

// If door just opened, start timing

else if (doorOpen && !doorWasOpen) {

doorOpenTime = millis(); // Record the time when door opened

}

// If door has been open, check if delay has passed and buzzer is not silenced

else if (doorOpen && !buzzerSilenced) {

if (millis() - doorOpenTime >= DOOR\_OPEN\_DELAY) {

controlBuzzer(true); // Sound the buzzer

Serial.println("Door open too long! Buzzer ON.");

} else {

controlBuzzer(false); // Wait until delay passes

}

}

// Remembers the last door state. So program detects door just opened.

doorWasOpen = doorOpen;

}