runlinc Project 3 AI1: Intruder Alarm (E32W Version)

Contents

Note: For the programming part of this project, there are two separate parts C1 and C2, which are corresponding to 2 different sets of codes for Apple devices and other devices.

Introduction

Problem

How can we use microchips to protect our valuables? How can a microchip know if there is a thief, and what can it do when it detects one?

Background

By learning E32W, you can learn to program microchips to tell them what to do. Microchips can monitor devices and warn people like when your laptop battery is low, and you need to plug the cable in. However, they can also be used to control cameras and even stream the camera to a webpage.

Ideas

Look at the E32W controller board. Can you see any inputs, i.e. something that we can touch or change to tell the microchip something? What about an output, i.e. something the microchip can change to tell us something? What kind of inputs and outputs are normally on an alarm system? What inputs and outputs can we use on our alarm system?

Plan

As we know, most alarms have some kind of flashing light along with some kind of an alert noise like sound or voice, as well as some kind or method of detection. To do this, we will use the lamp provided in the kit along with the light sensor for motion detection. For the voice, we will use the web browser's own speech. The lamp should flash for about 1 second on and 1 second off. The light sensor will have to be set to a threshold which can reduce excess sensitivity of the sensor.

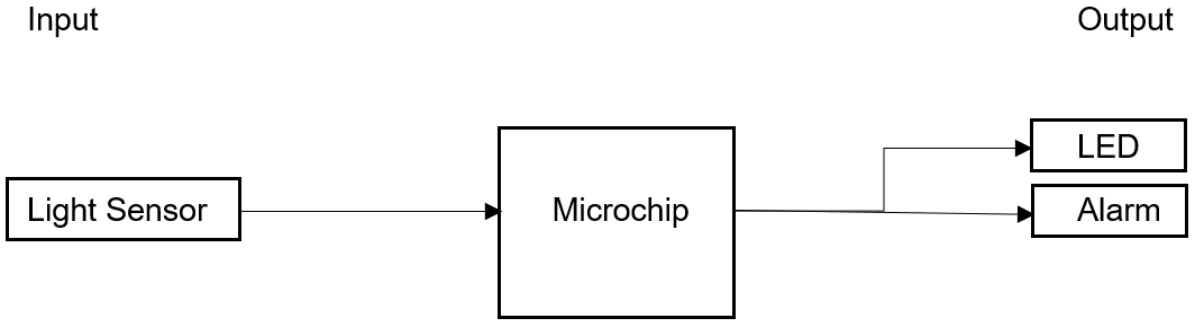

Figure 1: Block diagram of Microchip outputs

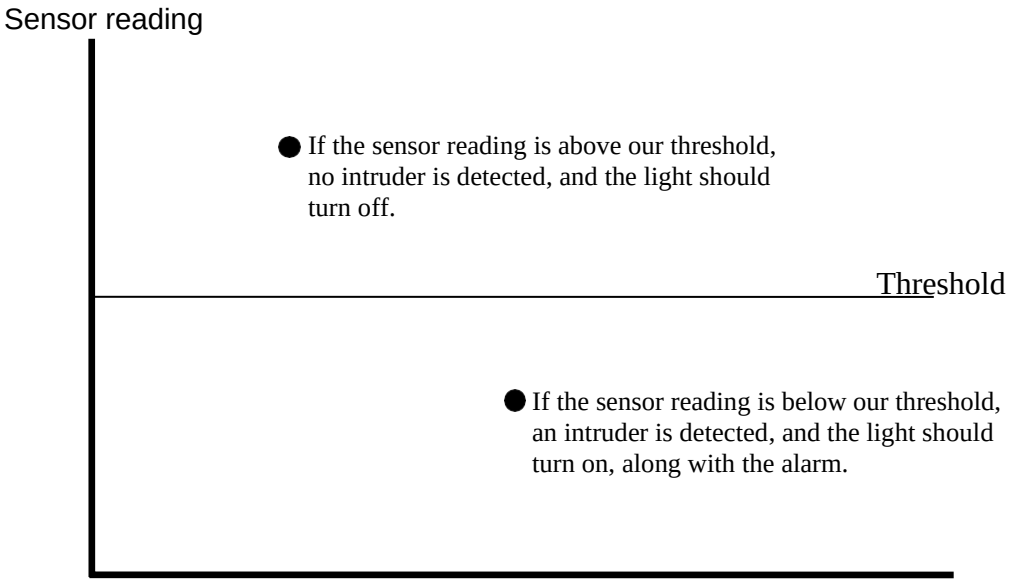

Figure 2: Light Level for a light sensor

runlinc Background

runlinc is a web page inside a Wi-Fi chip. The programming is done inside the browsers compare to programming inside a chip. The runlinc web page inside the Wi-Fi chip will command the microchips to do sensing, control, data logging Internet of Things (IoT). It can predict and command.

Part A: Design the Circuit on runlinc

Note: Refer to runlinc Wi-Fi Setup Guide document to connect to runlinc

Use the left side of the runlinc web page to construct an input/output (I/O).

For port D5 name it Red and set it as DIGITAL_OUT.
For port D18 name it Green and set it as DIGITAL_OUT.
For port D19 set it as DIGITAL_OUT (used at negative pin of LED – no name needed).
For port D33 name it LightSensor and set it as ANALOG_IN.

In our circuit design, we will be using a light sensor and LED Light. We happen to have these in our kits, so these can be used on our circuit design, as per the plan. Buzzer won't be used because it can't communicate. Instead, an AI voice generated from the web browser is used.

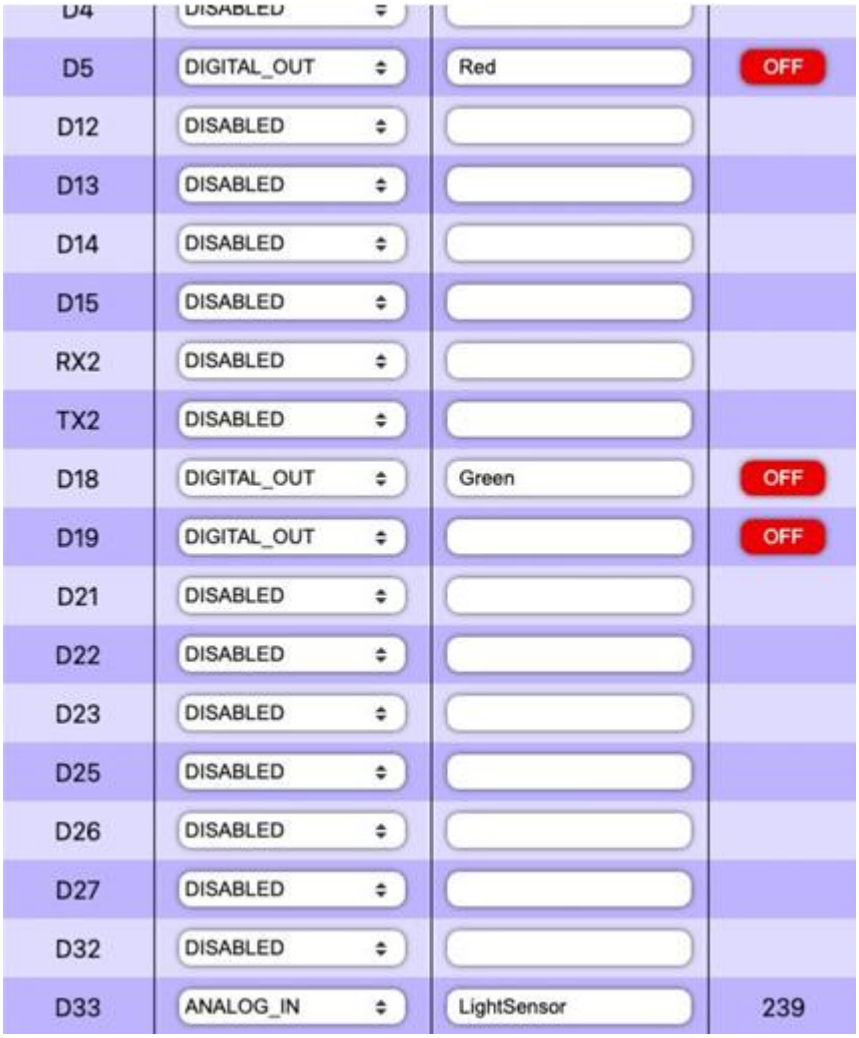

Figure 3: I/O configurations connections

Part B: Build the Circuit

Use the STEMSEL E32 board to connect the hardware. For this project we are using both the left and right I/O ports, with **negative port (-ve)** on the outer side, **positive port (+ve)** on the middle and **signal port (s)** on the inner side (as shown below).

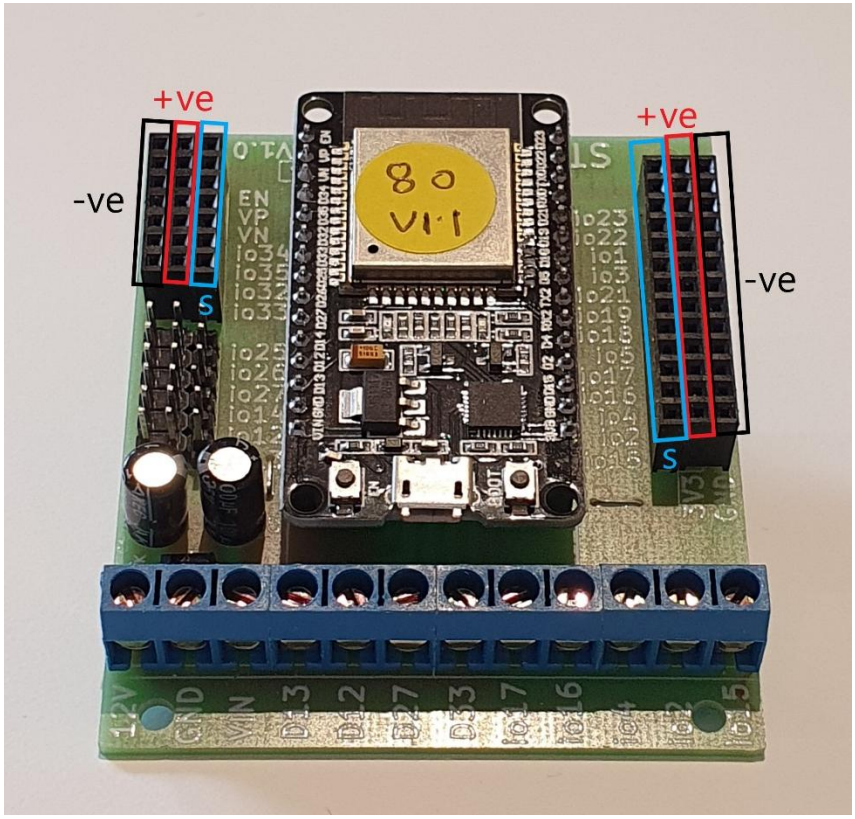

Figure 4: Negative, Positive and Signal port on the E32 board

There are two I/O parts we are using for this project, a 3-pin LED light and a Light Sensor, their respective pins are shown in the figure below.

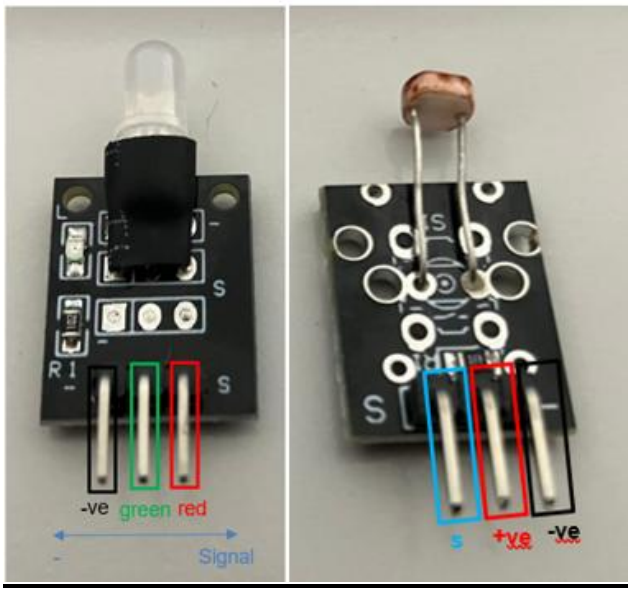

Figure 5: I/O parts with negative, positive and signal pins indicated

Wiring instructions

a.) Plug in the LED to io5, io18 and io19 (Signal "S" on io5) - on the signal pins near the chip on the E32 board.

b.) Plug in the Light Sensor to io33 on the E32 board.

c.) Make sure all (-ve) pins are on the GND (outer) side of the I/O ports.

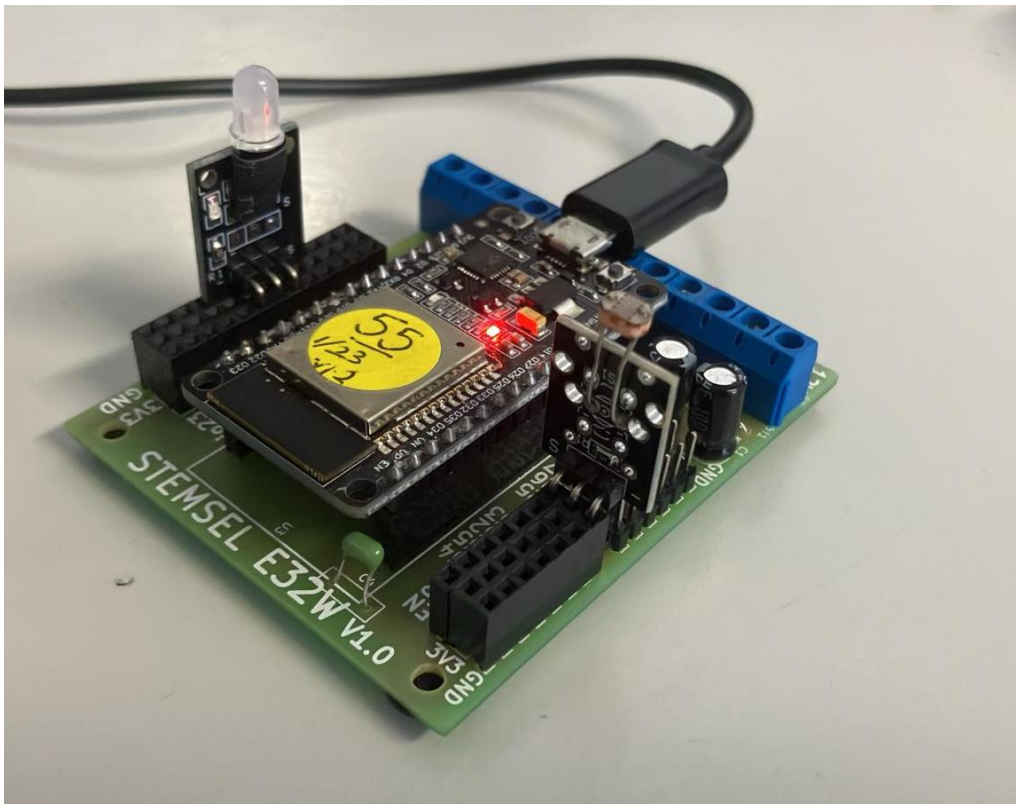

Figure 6: Circuit board connection with I/O parts (side view)

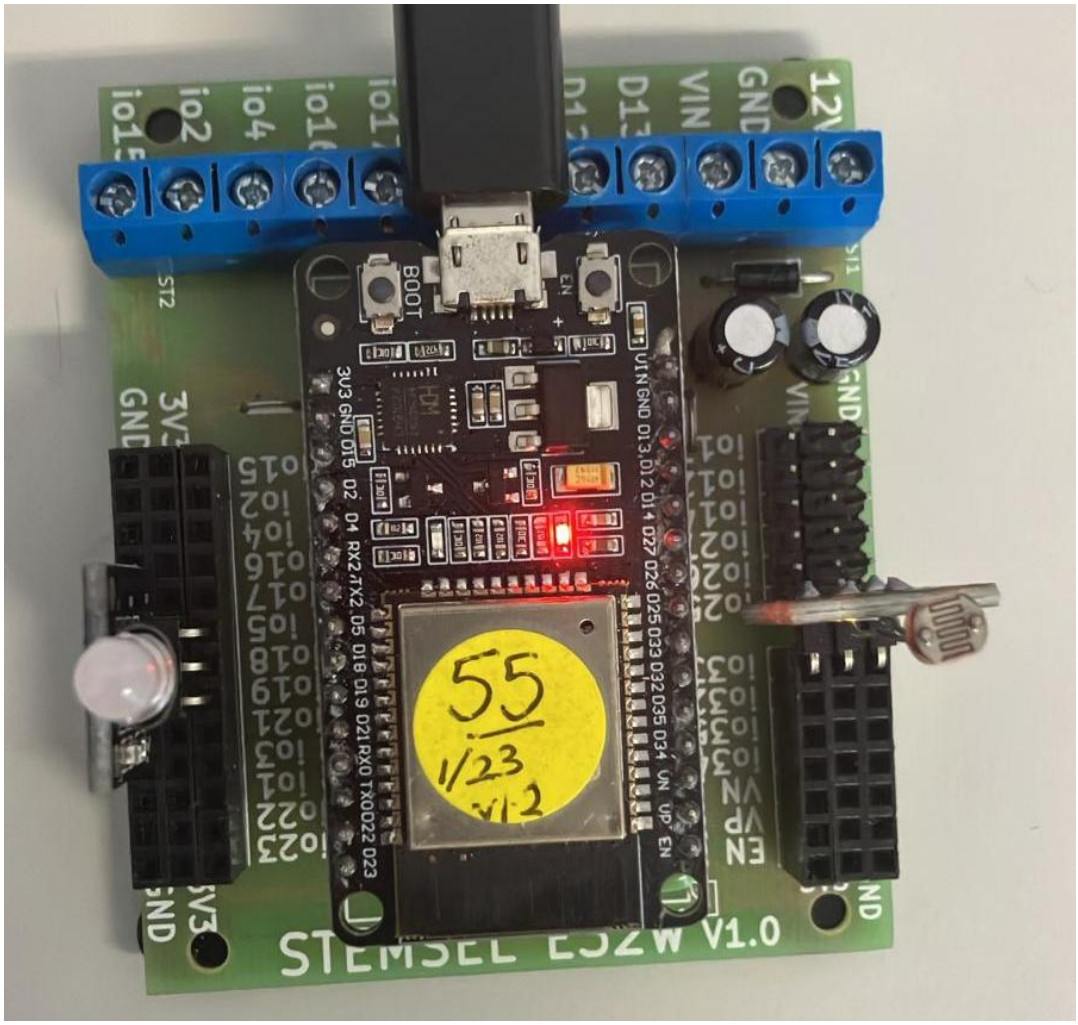

Figure 7: Circuit board connection with I/O parts (top view)

Part C1: Program the Circuit for Android & Windows

In JavaScript Loop, type in the following code:

```
if (analogIn(LightSensor) > 120){
 turnOff( Green );
 turnOn( Red );
 await mSec( 1000 );
 turnOff( Red );
 await mSec( 1000 );
```

This will create an LED which will turn red if the light sensor's reading is higher than 120. You may need to adjust the number to a number which is smaller than the biggest number that your light sensor can go up to when you cover the light sensor with your finger. Now that we have the flashing of the light setup, we need to give our alarm a voice. To do this, we will use speech synthesis. To do this follow the code snippet that follows:

```
const speech = new SpeechSynthesisUtterance("Intruder Alert");
 window.speechSynthesis.speak(speech);
}
```

This code will allow runlinc to access your browsers voice. Now that we have set up the voice, we need to make sure it doesn't go off when there is high enough light. To do this type in:

```
else{
 window.speechSynthesis.cancel();
 turnOff(Red);
 turnOn(Green);
}
```

Now add in a macro to turn off the light using the methods we used before. Make sure to close off your code with a "}"

In the end, you should have the program as below.

For **JavaScript Loop** box:

```
if (analogIn(LightSensor)>120){
 turnOff( Green );
 turnOn( Red );
 await mSec( 1000 );
 turnOff( Red );
 await mSec( 1000 );
```

```
 const speech = new SpeechSynthesisUtterance("Intruder Alert");
window.speechSynthesis.speak(speech);


}else{
 window.speechSynthesis.cancel();
 turnOff( Red );
 turnOn( Green );
}
```

Remember our Light Sensor have a higher reading when its surrounding is darker, and a lower reading when its surrounding is brighter.

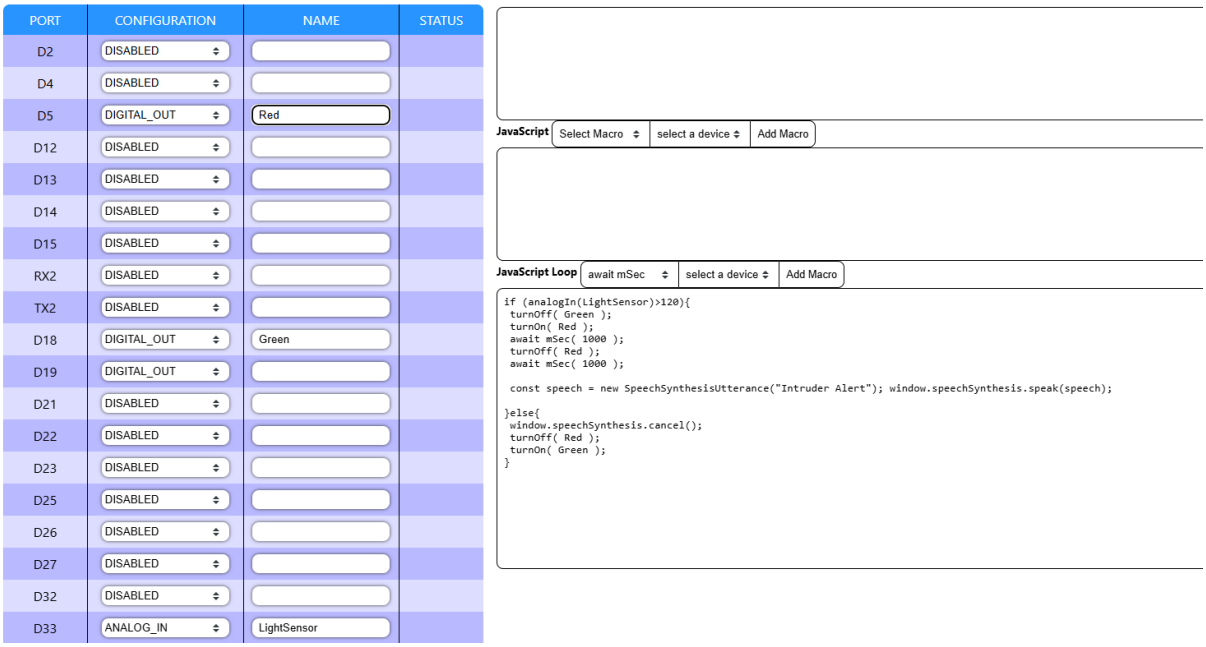

Figure 8: runlinc webpage screenshot

Part C2: Program the Circuit for Apple Devices

For Apple devices, the voice output needs to be activated by some trigger. We will use a button on the screen to do this.

First, in the HTML, type in following code to create a button:

```
<button style="font-size:40px;background-color:#90ee90;border-radius:0.5rem;"
id="startButtonID" onclick="buttonChangeText();">Start</button>
```

In JavaScript, type in the following code:

```
var stopSpeech = 0;
var InitialSetup = 0;
var allowFrequencyVariation = 1;
var allowAngleChecking = 0;
var alertInterval;
const audioContext = new (window.AudioContext || window.webkitAudioContext)();
```

This will create some variables for the functions you need to use in the following codes.

Next, we create a function, which is a piece of code which is wrapped in brackets and can be executed when being called by the code. This function will speak some words when it's been called. Let's name it startThisScript.

```
function startThisScript() {
  const utterance = new SpeechSynthesisUtterance("Initializing setup");
  window.speechSynthesis.speak(utterance);
}
```

Then we will write another function to use the light sensor, to turn on the red LED and say "Intruder Alert".

```
async function checkLightSensor() {
  if (analogIn(LightSensor) > 120) {
    turnOff(Green);
    turnOn(Red);
    await mSec(500);
    turnOff(Red);
    await mSec(500);
    const speech = new SpeechSynthesisUtterance("Intruder Alert");
    window.speechSynthesis.speak(speech);
  } else {
    window.speechSynthesis.cancel();
    turnOff(Red);
    turnOn(Green);
  }
}
```

At last, we write what the button will do:

```
async function buttonChangeText() {
  let IDButton = document.getElementById("startButtonID");

  if (IDButton.innerHTML == "Start") {
    stopSpeech = 0;
    IDButton.style = "font-size:40px;background-color:red;border-radius:0.5rem;";
    IDButton.innerHTML = "Stop";
    allowFrequencyVariation = 1;
    allowAngleChecking = 1;

    if (InitialSetup == 0) {
      startThisScript();
      InitialSetup = 1;
    }

    alertInterval = setInterval(checkLightSensor, 2000);

  } else {
    IDButton.style = "font-size:40px;background-color:#90ee90;border-radius:0.5rem;";
    IDButton.innerHTML = "Start";
    allowFrequencyVariation = 0;
    allowAngleChecking = 0;
    stopSpeech = 1;

    clearInterval(alertInterval);
    window.speechSynthesis.cancel();
  }
}
```

This will make the button change its colour to red and show “Stop”, trigger the program start sound, enable the voice output and check the reading of the light sensor every 2 seconds after it’s been clicked. If you click on it when its showing “Stop”, the button will turn to original text and colour, stop the speech and reset the program.

Summary

People can use programming to tell microchips what to do. However, sometimes those microchips, in turn, can warn people about dangers or intruders, so it is important to program them correctly. In this project, we learned that we can use lights in conjunction with a voice that can be used to achieve this.