runlinc Project 3 AI1:
Intruder Alarm (E32W Version)

Contents

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.

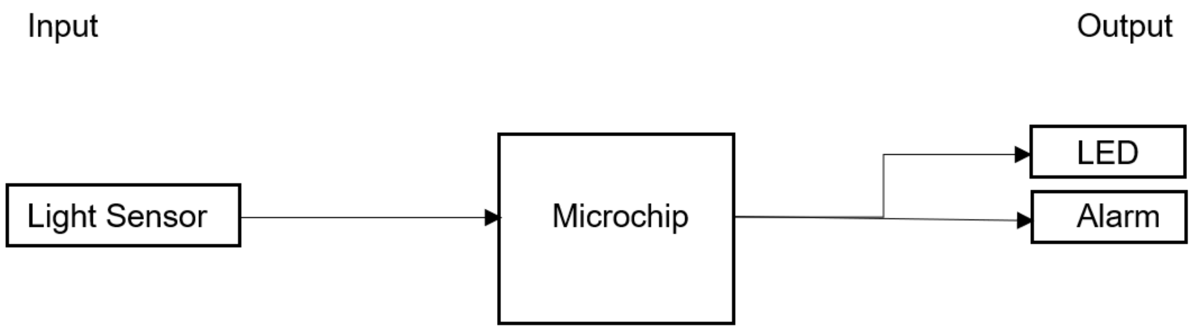


Figure 1: Block diagram of Microchip outputs

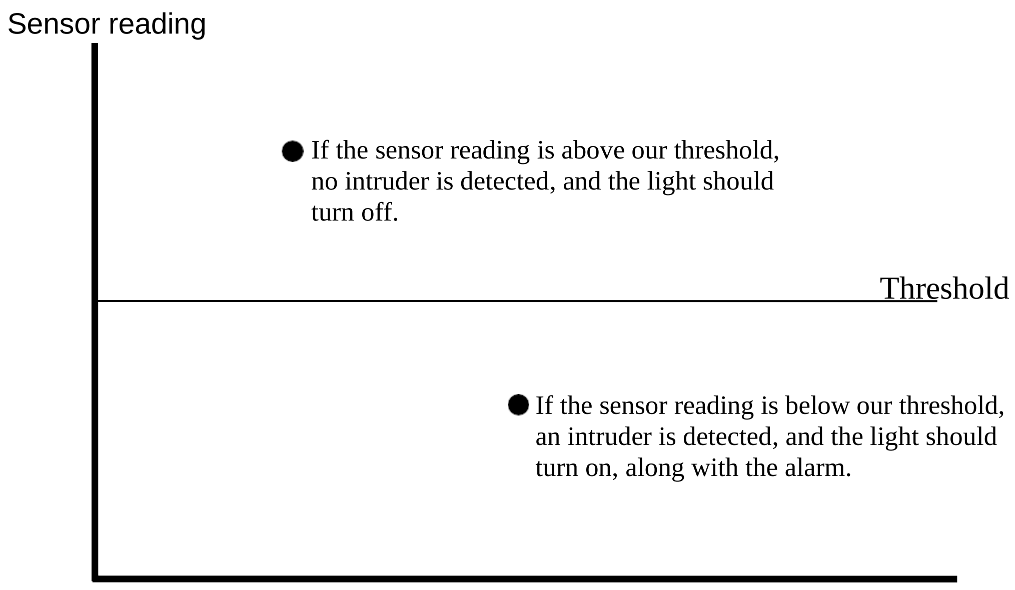

Sensor reading
If the sensor reading is above our threshold, no intruder is detected, and the light should turn off.
Threshold
If the sensor reading is below our threshold, an intruder is detected, and the light should turn on, along with the alarm.

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.

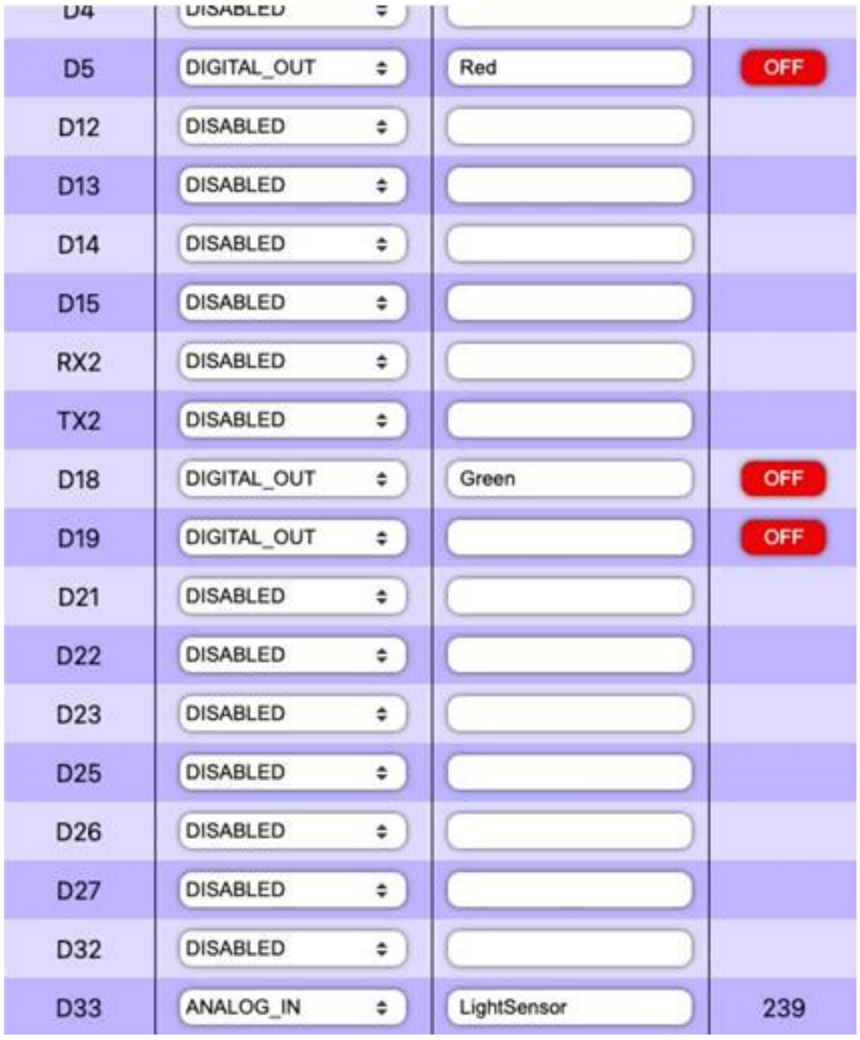
Part A: Design the Circuit on runlinc

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.



D4	DISABLED		
D5	DIGITAL_OUT	Red	OFF
D12	DISABLED		
D13	DISABLED		
D14	DISABLED		
D15	DISABLED		
RX2	DISABLED		
TX2	DISABLED		
D18	DIGITAL_OUT	Green	OFF
D19	DIGITAL_OUT		OFF
D21	DISABLED		
D22	DISABLED		
D23	DISABLED		
D25	DISABLED		
D26	DISABLED		
D27	DISABLED		
D32	DISABLED		
D33	ANALOG_IN	LightSensor	239

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).

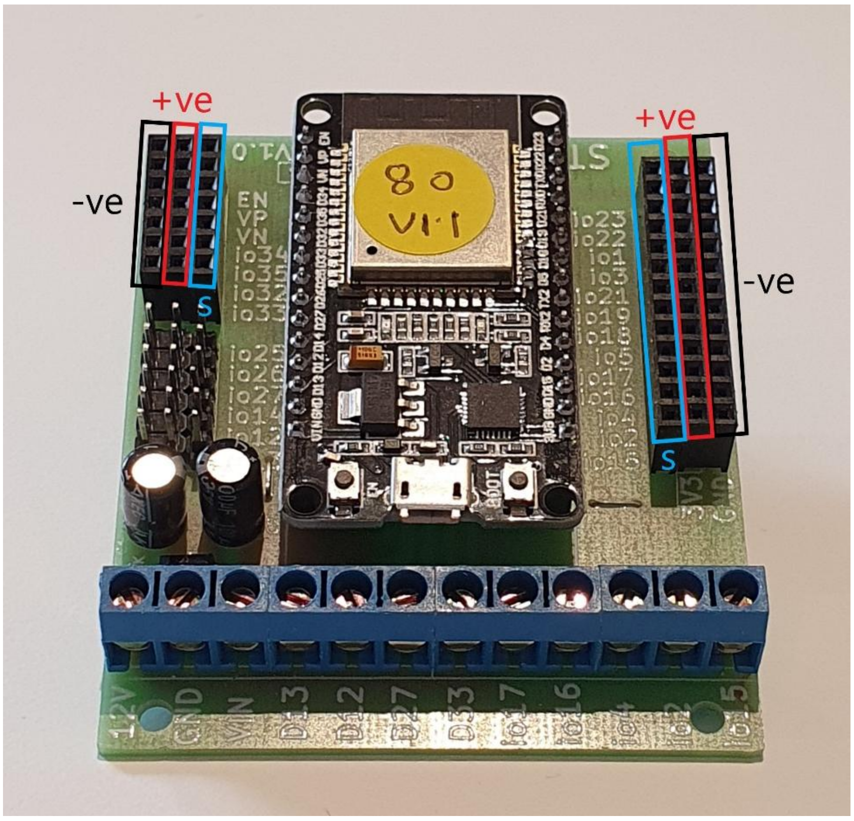


Figure 4: Negative, Positive and Signal port on the E32 board

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.

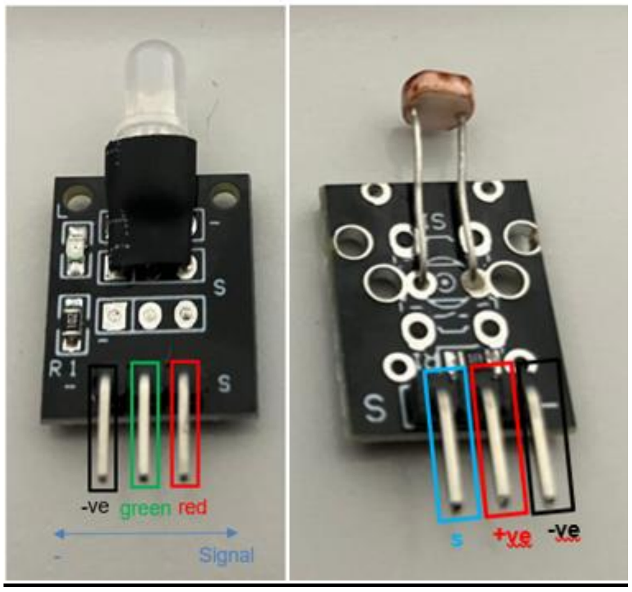


Figure 5: I/O parts with negative, positive and signal pins indicated

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.

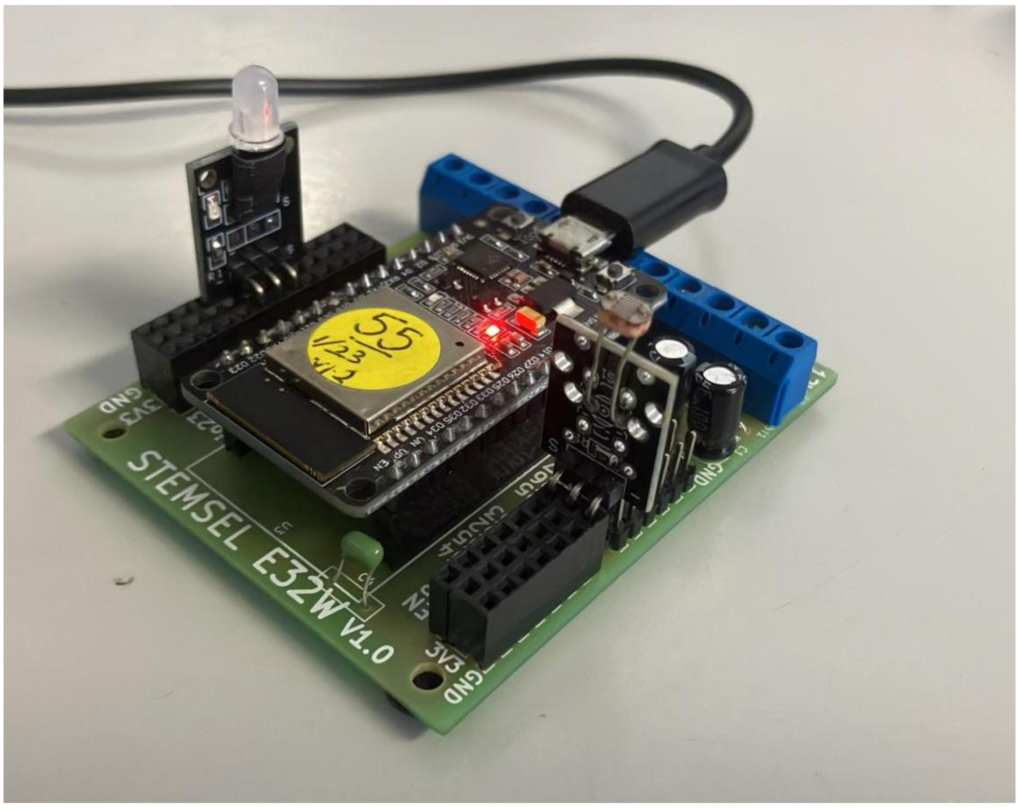


Figure 6: Circuit board connection with I/O parts (side view)

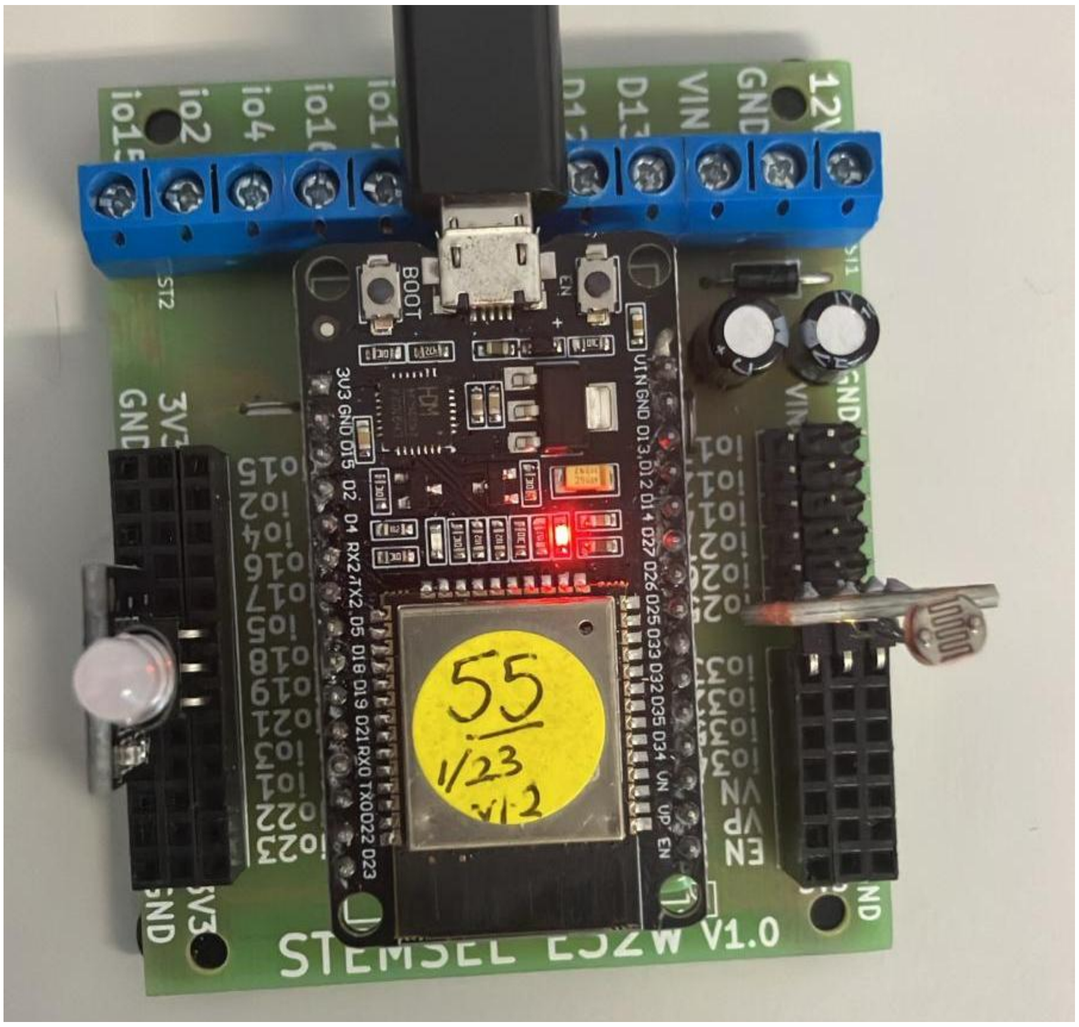


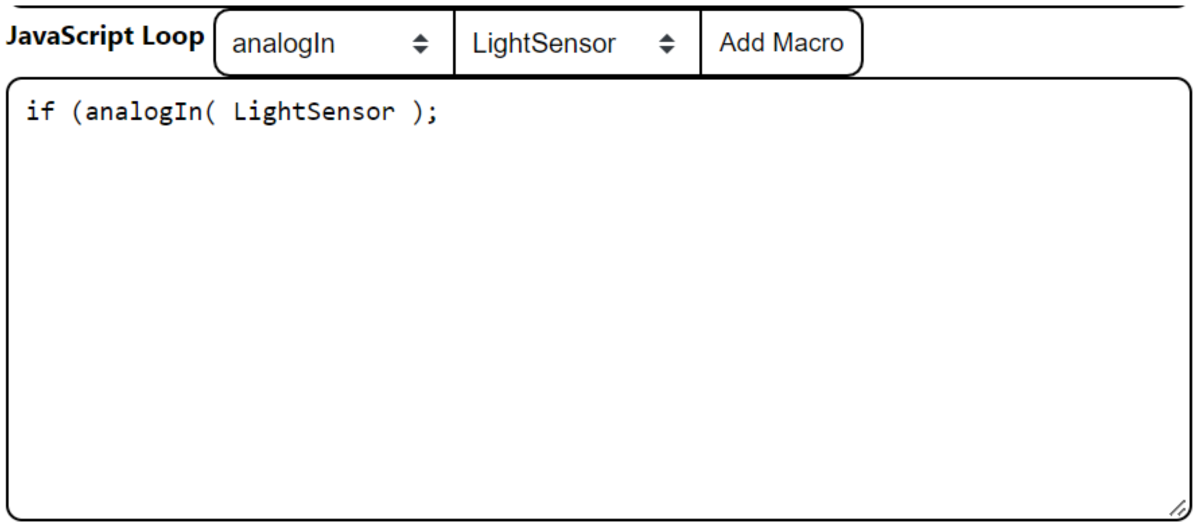
Figure 7: Circuit board connection with I/O parts (top view)

Part C: Program the Circuit

Use the blocks on the right side of the runlinc webpage to program the functions of the traffic light. Use the HTML to add contents, CSS to add style to your taste and Javascript to program the microchip. For this case, only JavaScript Loop is needed to program it to act as an alarm. Type the following code.

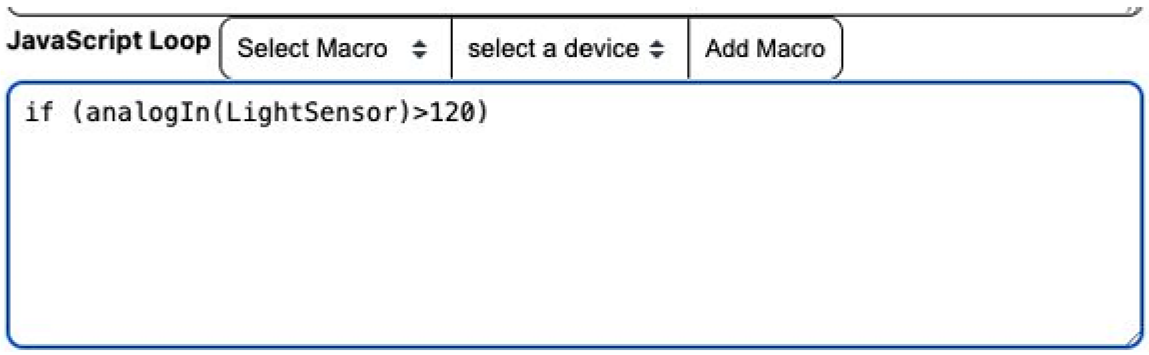
After naming ports D23 and D33, we are going to program the circuit. First, we need to declare an if statement. To do this, we will type into the JavaScript Loop "if (" then go to Select Macro and choose "analogIn" from the drop-down menu.

After that, go to the next button, **select a device**. Select the LightSensor.
And then click the **Add Macro** button to add the macro.



```
if (analogIn( LightSensor );
```

Now delete the semicolon and replace it with ">120". The 120 represents the threshold that everything (LED Light and AI Voice) will activate if it goes lower than it. Make sure the number works for your set-up; some might have to be lower or higher.



```
if (analogIn(LightSensor)>120)
```

Now that we have set up the if statement, we need to declare what it'll do if the numbers increase above 120. After the 120 add "{"

So next what we want to turn on our Red LED. In the JavaScript Loop, select the macro turnOff.

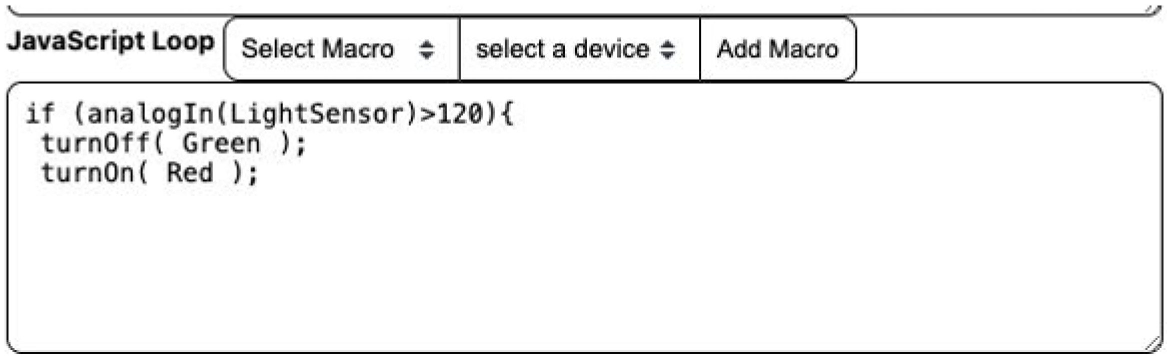
And then go to the next button, select a device. Select the Green.

Then add the macro.

Then select the macro turnOn.

And then go to the next button, select a device. Select the Red.

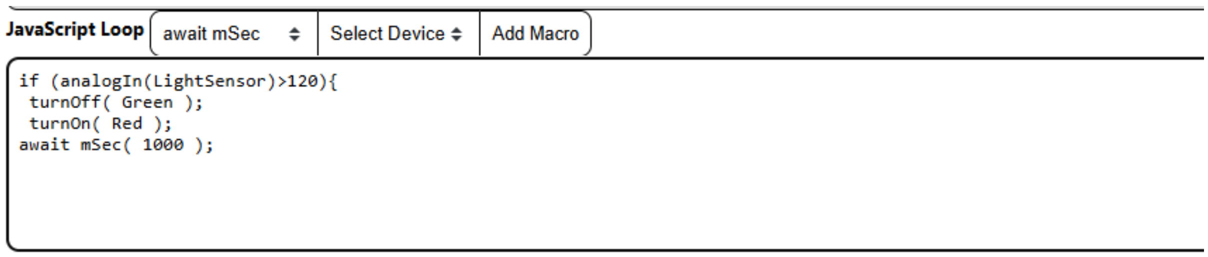
Then add the macro



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

Then, we need to keep the Light on for 1 second. Go to the **Select Marco** button, choose the **await mSec**.

And then click the **Add Macro**.

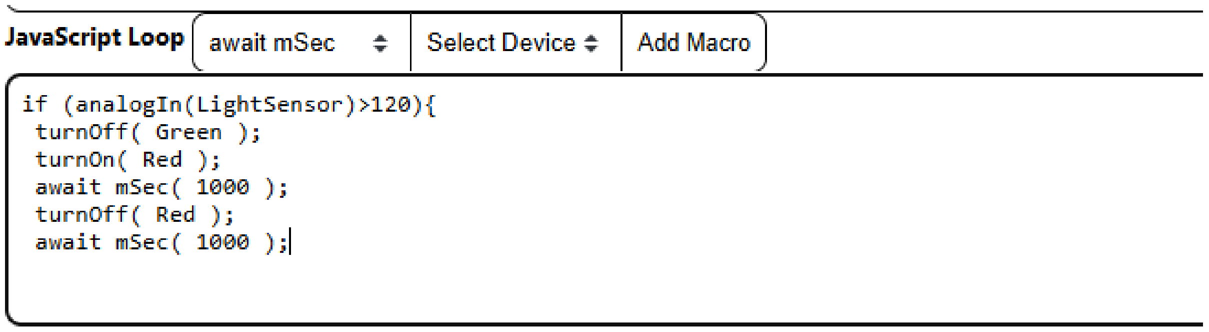


```
if (analogIn(LightSensor)>120){
 turnOff( Green );
 turnOn( Red );
await mSec( 1000 );
```

We will leave the 1000ms there as this will be the 1 sec that we want. After 1 sec, we want to turn off the red LED.

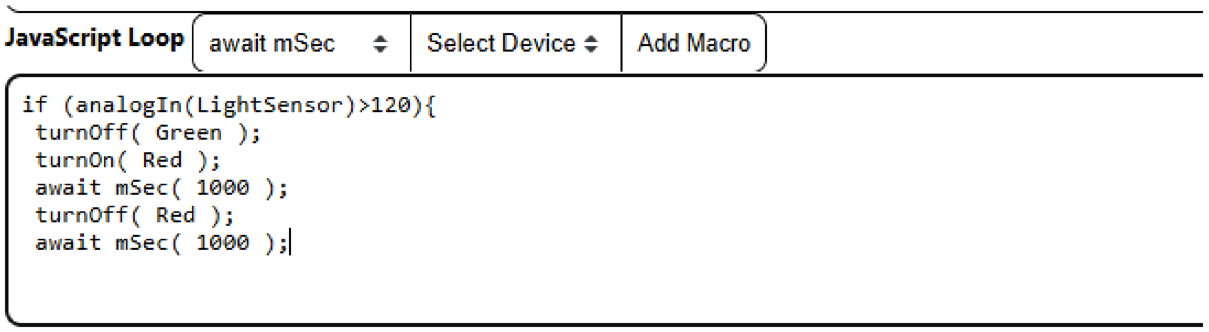
From the macro button, select turn off along with the Light under the devices.

Add the macro.



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

Now as this is a loop we will need to add another await mSec otherwise the light will never turn off. Go ahead and add another await for 1 sec.



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

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.

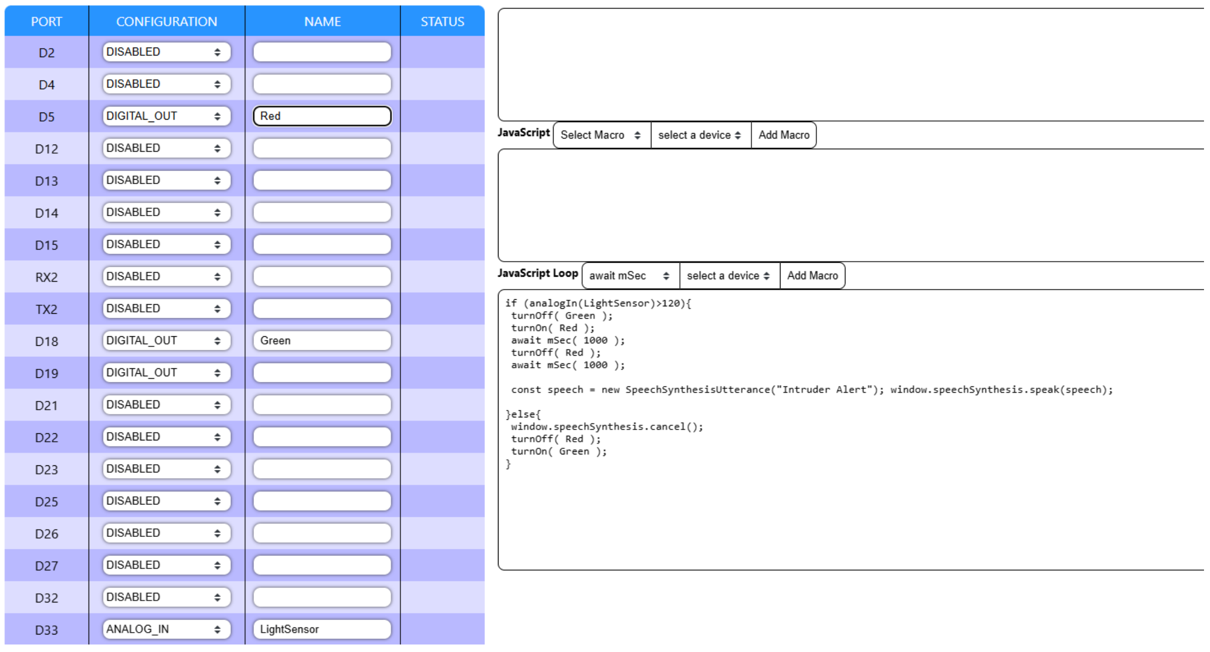
For **JavaScript Loop** box:

```
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.



PORT	CONFIGURATION	NAME	STATUS
D2	DISABLED		
D4	DISABLED		
D5	DIGITAL_OUT	Red	
D12	DISABLED		
D13	DISABLED		
D14	DISABLED		
D15	DISABLED		
RX2	DISABLED		
TX2	DISABLED		
D18	DIGITAL_OUT	Green	
D19	DIGITAL_OUT		
D21	DISABLED		
D22	DISABLED		
D23	DISABLED		
D25	DISABLED		
D26	DISABLED		
D27	DISABLED		
D32	DISABLED		
D33	ANALOG_IN	LightSensor	

```
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 );
}
```

Figure 8: runlinc webpage screenshot

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.