



# runlinc AI Project 1: Intruder Voice Synthesiser (STEMSEL Version)

---

## Contents

Introduction .....	1
Part A: Design the Circuit on runlinc.....	3
Part B: Build the Circuit .....	3
Part C: Program the Circuit.....	4
Summary .....	9

## 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

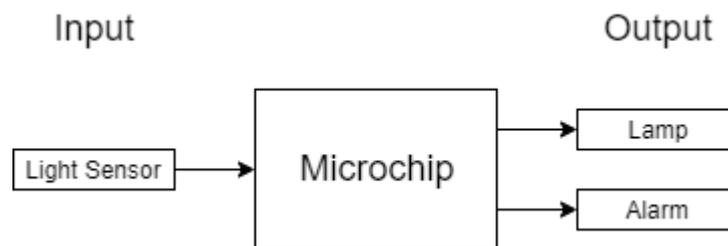
By learning STEMSEL, you can learn to program microchips to tell them what to do. Microchips can monitor devices and warn people like when your laptop battery is low, and you need to plug the cable in. However, they can also be used to control cameras and even stream the camera to a webpage.

### Ideas

Look at the STEMSEL 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 browsers 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*

## 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

In our circuit design, we will be using the Lamp and a Light Sensor. We happen to have these in our kits, so these can be used on our circuit design, as per the plan.

On the runlinc webpage remember to type in a name for the equipment we are using

For port C7, name it LightSensor and set it as ANALOG\_IN

For port C4, name it Light and set it as DIGITAL\_OUT

runlinc V1.1 Copyright and International Patent Pending. All rights reserved

PORT	CONFIGURATION	NAME	STATUS
A3	DISABLED		
B4	DISABLED		
B6	DISABLED		
C0	DISABLED		
C1	DISABLED		
C2	DISABLED		
C3	DISABLED		
C4	DIGITAL_OUT	Light	OFF
C5	DISABLED		
C6	DISABLED		
C7	ANALOG_IN	LightSensor	107

Figure 2 I/O configurations connections

## Part B: Build the Circuit

Use the runlinc I/O to connect the hardware. Remember that black wires connect to the negative port (-), red wires to the positive port (+) and white wires connect to the pin designated in the circuit design.

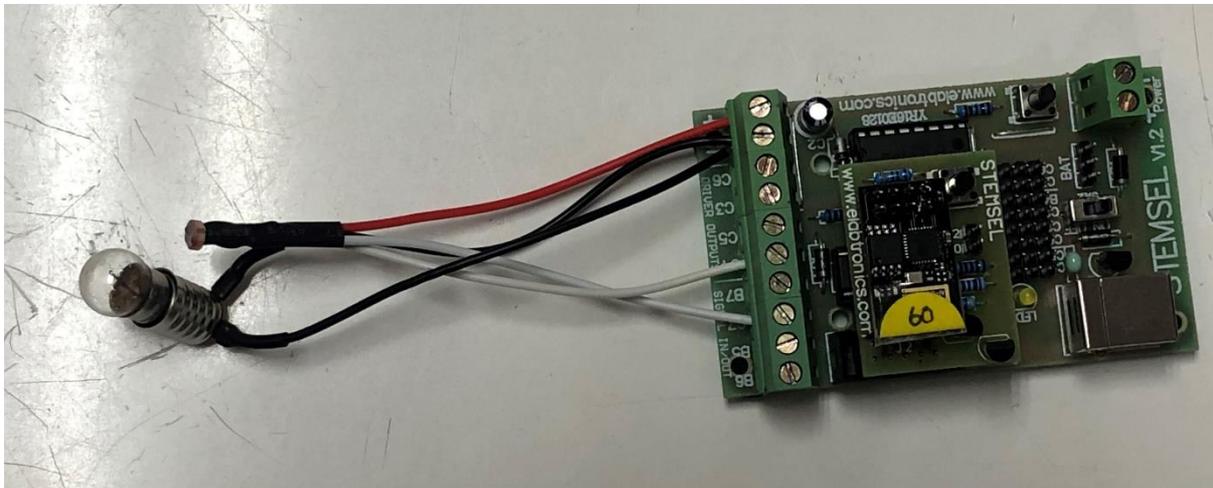


Figure 3: Circuit connection with microchip

- Connect the white wire of the LightSensor to C7
- Connect the white wire of the Lamp to C4
- Connect all the black wires to the negative port (-)
- Connect the red wire of the LightSensor to the positive port (+)

## 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 C4 and C7, 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.

The screenshot shows the JavaScript Loop configuration interface. The 'Select Macro' dropdown menu is open, displaying a list of options: turnOn, turnOff, analogIn (highlighted), digitalIn, await mSec, infinite loop, function, async function, if, else if, else, and while. The code editor area contains the text 'if ('.

After that, go to the next button, **select a device**. Select the LightSensor.

The screenshot shows the JavaScript Loop configuration interface. The 'select a device' dropdown menu is open, displaying a list of options: Light (highlighted) and LightSensor. The 'analogIn' text is visible in the 'Select Macro' dropdown.

And then click the **Add Macro** button to add the macro.

JavaScript Loop analogIn LightSensor Add Macro

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

Now delete the semicolon and replace it with "<160". The 160 represents the number that everything 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.

JavaScript Loop analogIn LightSensor Add Macro

```
if (analogIn( LightSensor )<160
```

Now that we have set-up the if statement, we need to declare what it'll do if the numbers drop below 160. After the 160 add "{"

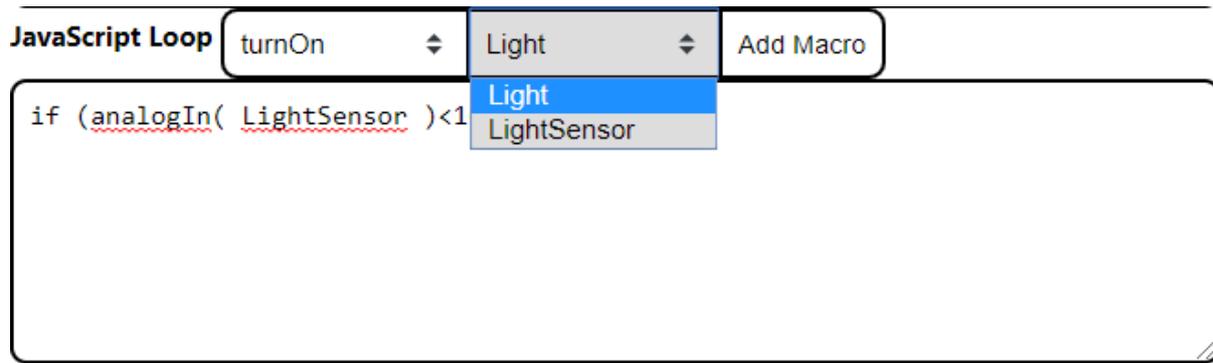
So next what we want to turn on our light. In the JavaScript Loop, select the macro turnOn.

JavaScript Loop turnOn Light Add Macro

```
if (analogIn( 60
```

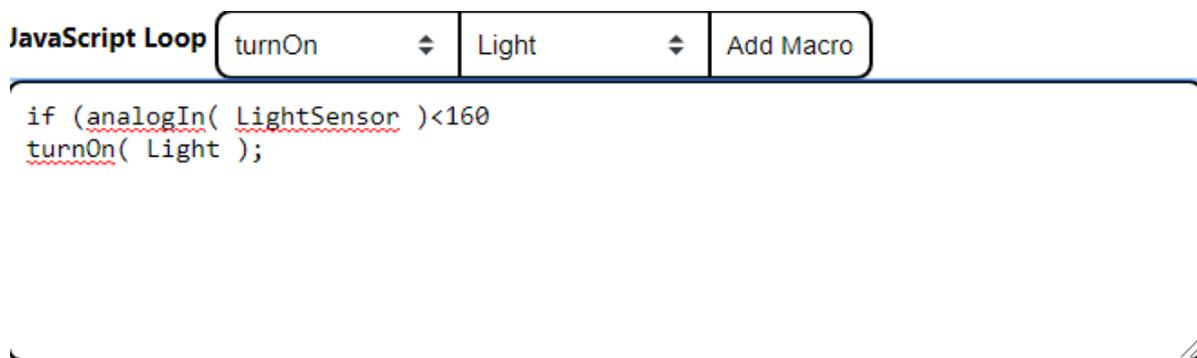
- turnOn
- turnOff
- analogIn
- digitalIn
- await mSec
- infinite loop
- function
- async function
- if
- else if
- else
- while

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



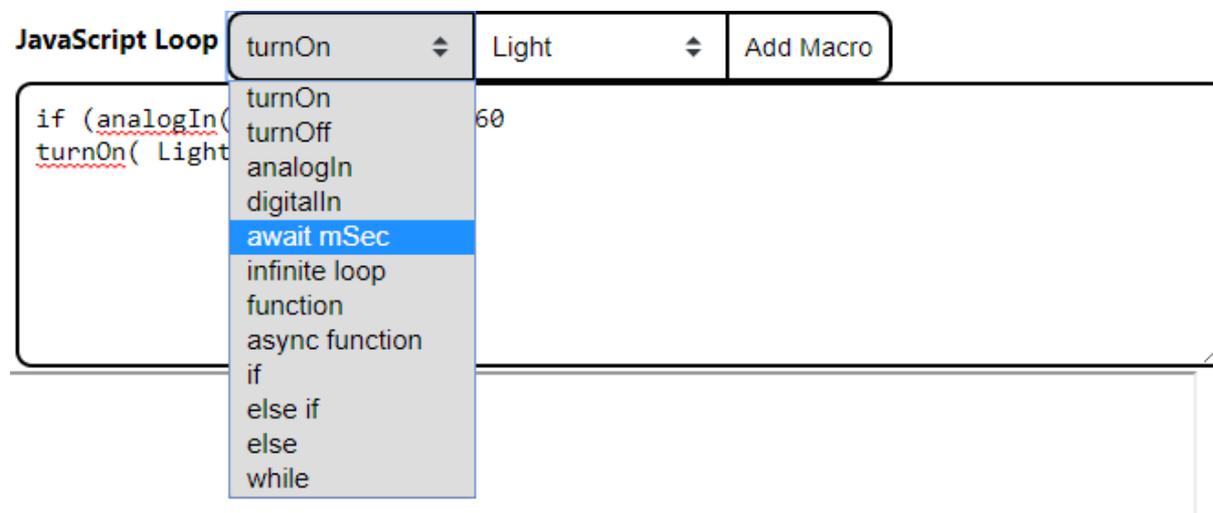
The screenshot shows the 'JavaScript Loop' editor. At the top, there are three buttons: 'turnOn', 'Light', and 'Add Macro'. The 'Light' button is selected, and a dropdown menu is open below it, showing 'Light' and 'LightSensor'. The code in the editor is: `if (analogIn( LightSensor ) < 1`

Then add the macro.



The screenshot shows the 'JavaScript Loop' editor. The 'turnOn' and 'Light' buttons are still present. The code in the editor is: `if (analogIn( LightSensor ) < 160  
turnOn( Light );`

Then, we need to keep the Lamp on for 1 sec. Go to the **Select Macro** button, choose the **await mSec**.



The screenshot shows the 'JavaScript Loop' editor. The 'Add Macro' button is selected, and a dropdown menu is open below it, showing various macros: 'turnOn', 'turnOff', 'analogIn', 'digitalIn', 'await mSec', 'infinite loop', 'function', 'async function', 'if', 'else if', 'else', and 'while'. The 'await mSec' macro is selected. The code in the editor is: `if (analogIn( LightSensor ) < 160  
turnOn( Light );`

And then click the **Add Macro**.

JavaScript Loop    await mSec    Light    Add Macro

```
if (analogIn( LightSensor )<160
turnOn( Light );
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 light. From the macro button, select turn off along with the Light under the devices.

JavaScript Loop    turnOff    Light    Add Macro

```
if (analogIn(
turnOn( Light
await mSec( 1
```

- turnOn
- turnOff
- analogIn
- digitalIn
- await mSec
- infinite loop
- function
- async function
- if
- else if
- else
- while

Add the macro.

JavaScript Loop    turnOff    Light    Add Macro

```
if (analogIn( LightSensor )<160
turnOn( Light );
await mSec( 1000 );
turnOff( Light );
|
```

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.

**JavaScript Loop**    await mSec    Light    Add Macro

```

if (analogIn( LightSensor )<160
turnOn( Light );
await mSec( 1000 );
turnOff( Light );
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();
}

```

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** the code is:

```

if (analogIn(LightSensor)<100){
    turnOn( Light );
    await mSec( 1000 );
    turnOff( Light );
    await mSec( 1000 );
    const speech = new SpeechSynthesisUtterance("Intruder Alert");
    window.speechSynthesis.speak(speech);
}else{
    window.speechSynthesis.cancel();
    turnOff( Light );
}

```

## 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.

**runlinc V1.1** Copyright and International Patent Pending. All rights reserved.

**File**

Load File  Save

Run Code  Stop Code

**Board**

Send  Get

Board IP:

**CSS**

**STEMSEL**

PORT	CONFIGURATION	NAME	STATUS
A3	DISABLED	<input type="text"/>	
B4	DISABLED	<input type="text"/>	
B6	DISABLED	<input type="text"/>	
C0	DISABLED	<input type="text"/>	
C1	DISABLED	<input type="text"/>	
C2	DISABLED	<input type="text"/>	
C3	DISABLED	<input type="text"/>	
C4	DIGITAL_OUT	Light	OFF
C5	DISABLED	<input type="text"/>	
C6	DISABLED	<input type="text"/>	
C7	ANALOG_IN	LightSensor	81

Network Status: Active

**HTML**

**JavaScript** Select Macro select a device Add Macro

**JavaScript Loop** turnOff Light Add Macro

```

if (analogIn(LightSensor)<100){
  turnOn( Light );
  await mSec( 1000 );
  turnOff( Light );
  await mSec( 1000 );
  const speech = new SpeechSynthesisUtterance("Intruder Alert");
  window.speechSynthesis.speak(speech);
}else{
  window.speechSynthesis.cancel();
  turnOff( Light );
}
                
```

Figure 4: runlinc webpage screenshot