runlinc Beginners Project 4: Control Buttons (STEMSEL Version)

Contents

Introduction 1
Part A: Design the Circuit on runlinc 3
Part B: Build the Circuit 4
Part C: Program the Circuit 5
Challenges 8
Summary 8

Introduction

Problem

Make HTML controlling buttons that can control a water sprinkler for watering your yard by using runlinc and the STEMSEL chip.

Background

By learning STEMSEL, you can learn how to program microchips and tell them what to do. Microchips can monitor devices and warn people like when your laptop battery is low, it will tell you that you need to plug the cable in. However, they can also control watering systems for yards or greenhouses even to help protect a property from a bushfire. But the microchips must send the right message, otherwise there can be some problems like turning on the watering system at the wrong times.

Ideas

As we can't control an actual watering system, what kind of things could we use instead that comes with the STEMSEL kit? What outputs does the chip have? What should we turn on first? Should we be able to turn them off?

Plan

We will use a yellow LED to represent that the watering system is preparing to come on and a buzzer will indicate that the water is running.

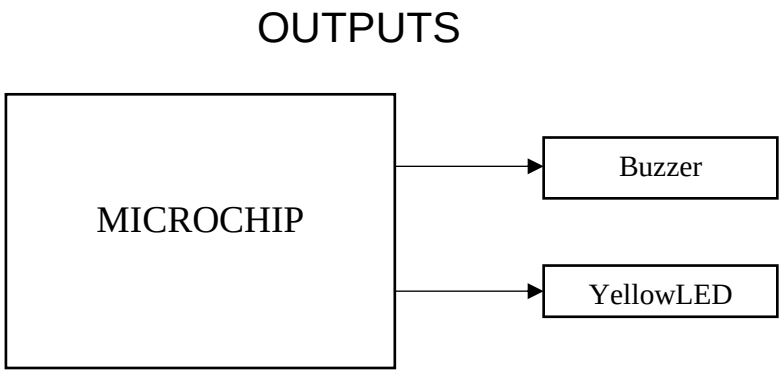

Figure 1: Block diagram of Microchip outputs

runlinc Background

Runlinc is a web page inside a Wi-Fi chip. The programming is done in the browser and sent to the chip over Wi-Fi. 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 a **yellow** LED and a buzzer. We happen to have both of them in our kits, so these can be used on our circuit design, as per the plan.

On the runlinc webpage, remember to type in the correct colour for the LED

For port C4, name it Buzzer and set it as DIGITAL_OUT

For port C5, name it YellowLED and set it as DIGITAL_OUT

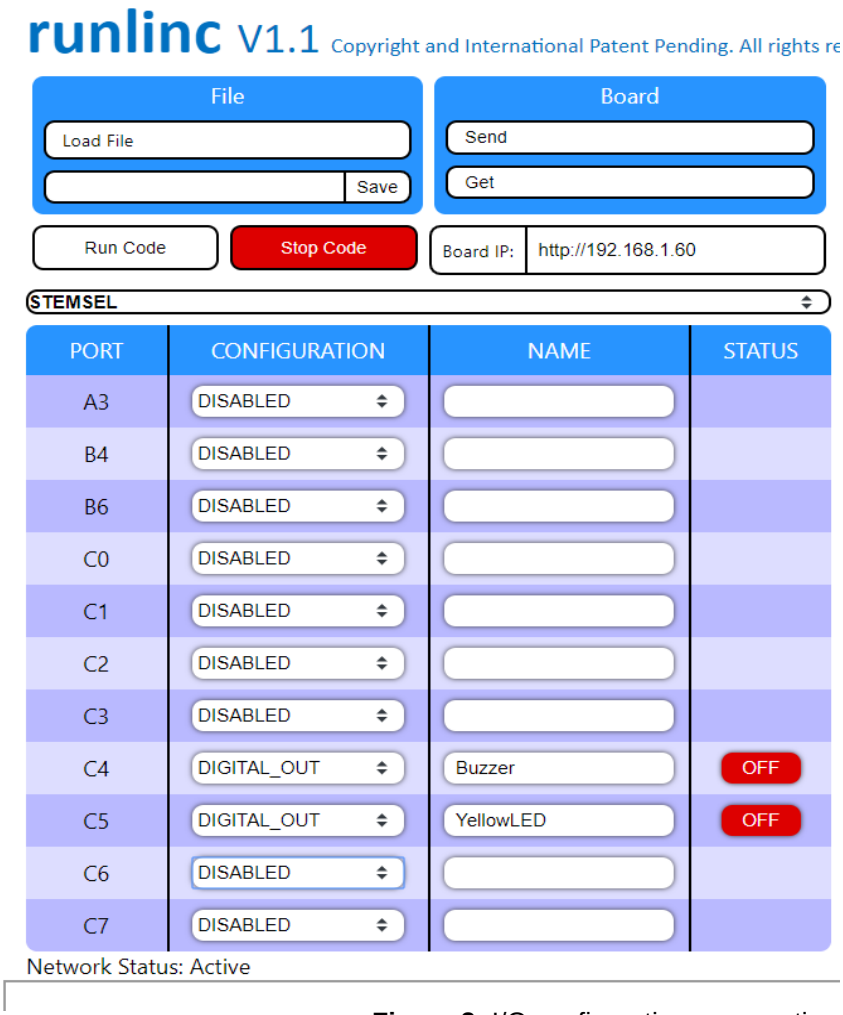

Figure 2: I/O configurations connections

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.

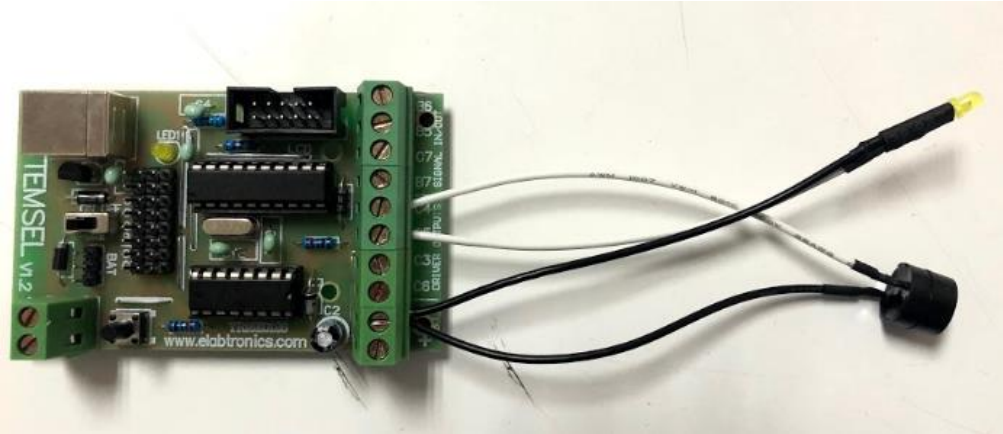

Figure 3: Circuit connection with microchip

Wiring instructions

a.) Connect the white wire of Buzzer to C4
b.) Connect the white wire of Yellow LED to C5
c.) Connect all the black wires to the negative port (-)

Part C: Program the Circuit

Use the blocks on the right side of the runlinc webpage to program the functions of the control buttons. Use the HTML to add contents, CSS to add style in your favour and Javascript to program the microchip. In this case, only the HTML is needed to program the buttons.

After naming ports C4 and C5, we are going to program the circuit. Firstly, we will create a header for our web page. To do this, we will use the h1 tag <h1> followed by the name of your page. Then use the closing version of the tag </h1>. It should look like below:

```
<h1>My Control Web Page</h1>
```

Once done push enter to go to the next line and type in YellowLED. This will simply label our buttons, so we know what they do. Then go to the next line.

In this line, we will create the button to turn on the YellowLED. To do this will we use the button and onclick method to create the button. Start by using this code to get you going:

```
<button onclick = “turnOn(YellowLED)”> ON </button>
```

Remember when coding that the code is case sensitive, so make sure that you have capital letters when needed, like with the “turnOn” statement.

Now you can run the code and see what happens. What’s wrong with the code currently? How could we solve it?

We need to create the off button for the YellowLED. Now we could type it in again as it is almost identical to the code we did above, or we could simply copy and paste the code on a new line and change the sections that need changing.

```
<button onclick = “turnOff(YellowLED)”> OFF </button>
```

After typing this section, you can add a
 tag to separate the next buttons

Now we can create the buzzer buttons in the same way we created the YellowLED buttons. So start with the name of the buzzer that will sit above the buttons using a
 tag after.

Now we create the buttons:

```
<button onclick = “turnOn(Buzzer)”>ON</button>
<button onclick = “turnOff(Buzzer)”>OFF</button>
```

After finishing and running the code, make sure to save your code and give it a name. Now we can send the code to the chip. To do this, click the “send” button. You will get two popups, this will indicate that the code has been successfully sent to the chip.

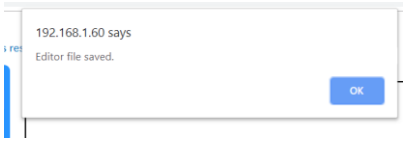

Figure 4: Editor file Saved

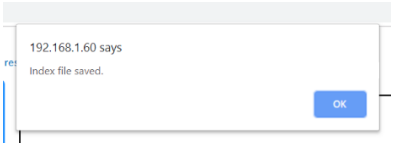

Figure 5: Index file saved

In the address bar at the top, remove the “/control.html” and push enter. This will take you to the web page that you have just created where you’ll be able to use the buttons to control your chip.

Now go back to the “/control.html” site. You’ll notice that it’s blank, you can either load the file from your system or you can click the button “get”, this will retrieve your code from the chip.

For **HTML** the code is:

```
<h1> My control page of awesomeness</h1>
YellowLED <br>
<button onclick="turnOn(YellowLED)">ON</button>
<button onclick="turnOff(YellowLED)">OFF</button><br>
Buzzer <br>
<button onclick="turnOn(Buzzer)">ON</button>
<button onclick="turnOff(Buzzer)">OFF</button>
```

The following is the expected result:

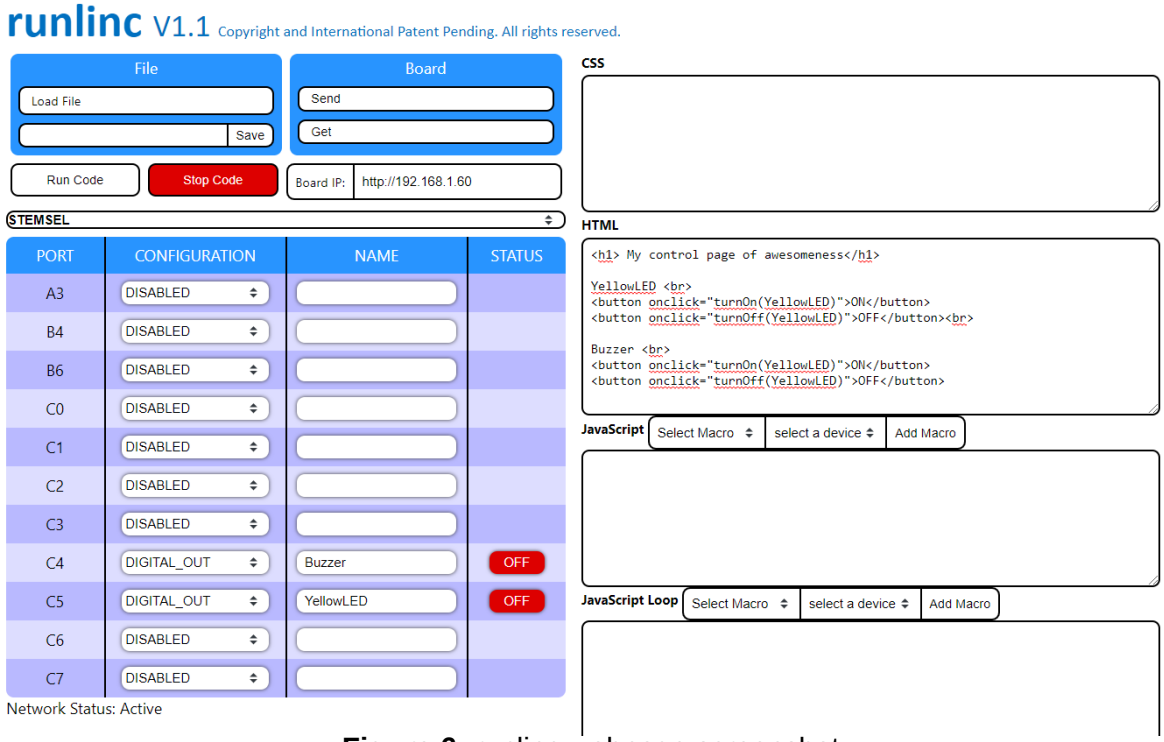

Figure 6: runlinc webpage screenshot

Figure 7: Web page Screenshot

Challenges

Now that the main code is done we can start to have a look at what else we can do with this. Take a look at your kit and see what else you can add and give them some new buttons to turn them on and off.

Maybe we can have a look at changing some of the styles of text or even the background colours of the buttons. To get you started here is a sample of some code to help.

<style = “color:tomato”>

You can also have a look at w3schools for other ideas and what to do.

Summary

People can use programming to tell microchips what to do. We can use them to turn on and off equipment, send signals to each other, all using buttons on HTML.