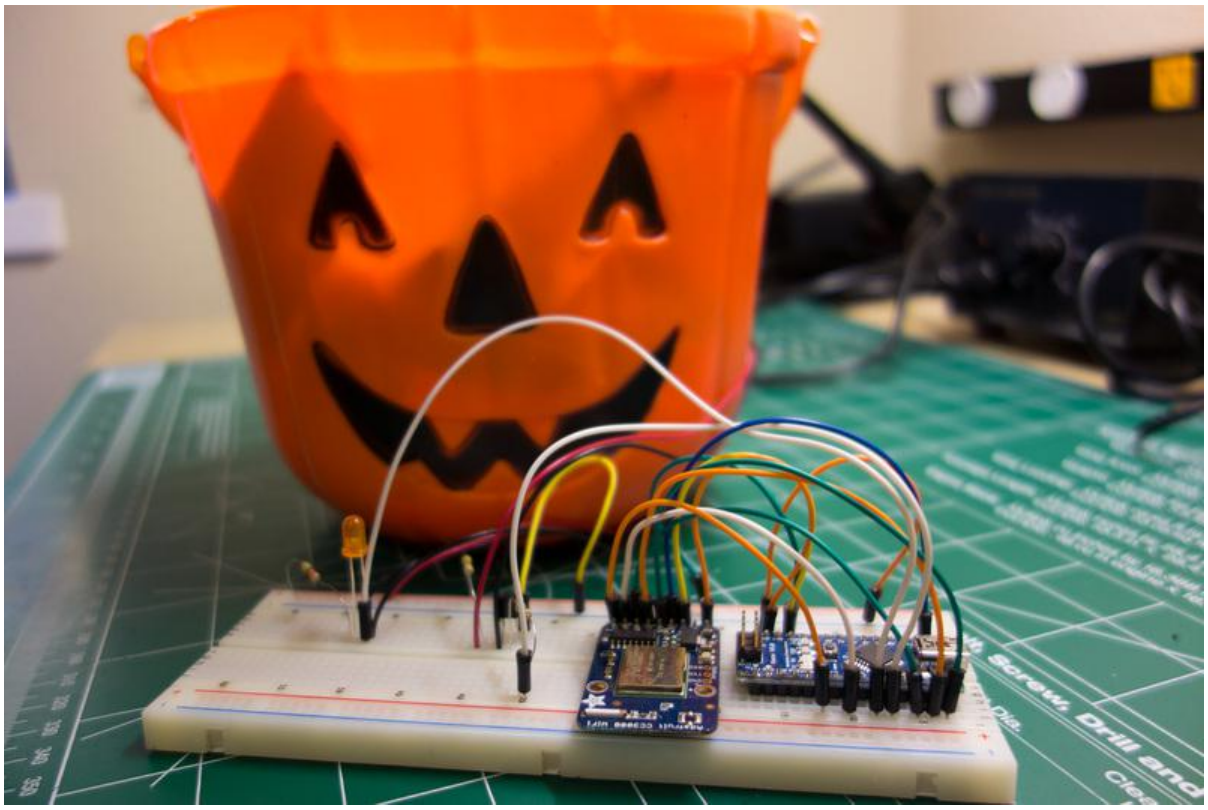
WiFi Candy Bowl Monitor

Created by Tony DiCola



https://learn.adafruit.com/wifi-candy-bowl

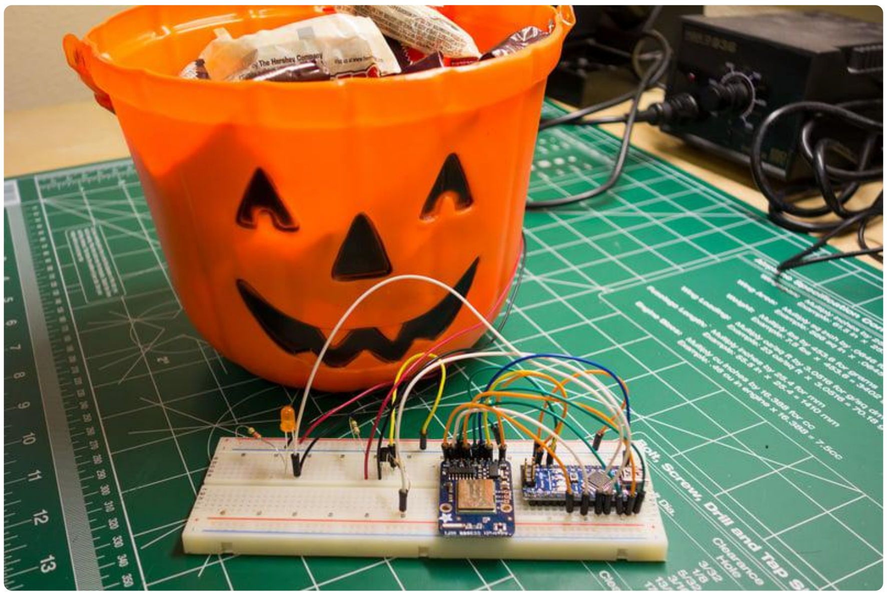
Last updated on 2022-12-01 02:04:25 PM EST

Table of Contents

Overview

Build this candy bowl you can monitor remotely over a WiFi network so trick-or-treaters never leave your home empty handed. A simple infrared light sensor detects when the bowl is empty or full, and a CC3000 WiFi chip exposes the sensor data to your wireless network. You can telnet to a simple server running on the Arduino and ask it if the bowl is full of candy!

This tutorial is a great demo of using the CC3000 breakout/shield in server mode, and also with mDNS so you don't need to know the IP address of the module!



Background

Before you get started it will help to familiarize yourself with the CC3000 and IR sensors by reading these related guides:

- Adafruit CC3000 WiFi ()
- IR Sensor ()

Continue on to learn about the hardware for this project.

Hardware

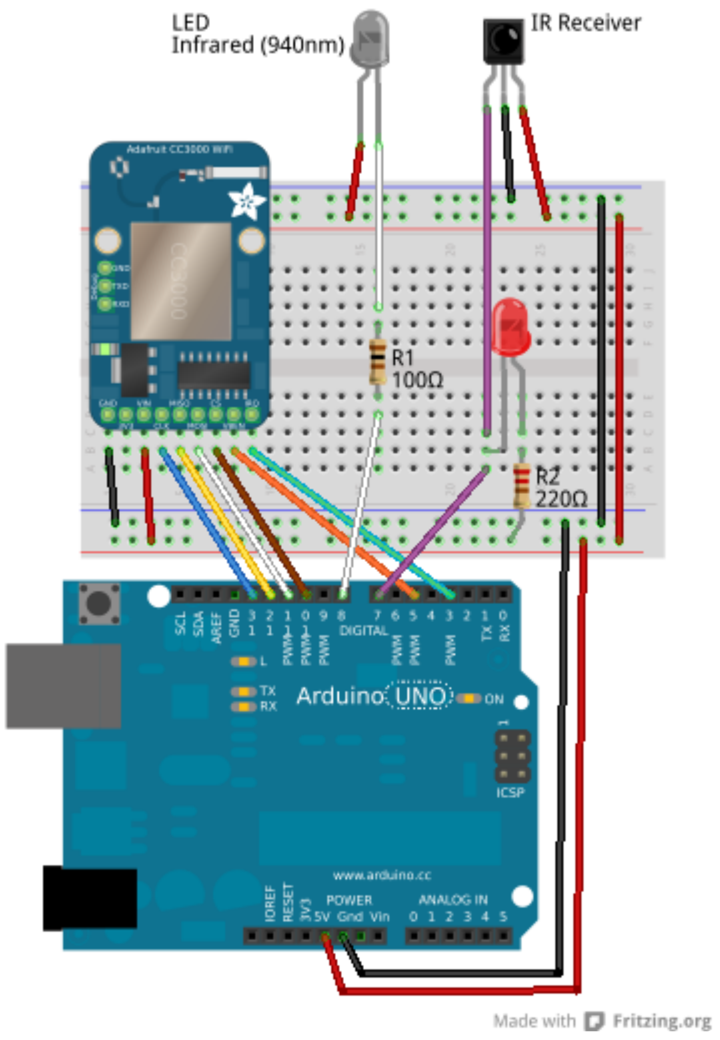
Parts

To build this project you'll need the following hardware and tools:

- Arduno Uno (http://adafru.it/50), Mega (http://adafru.it/191), or Nano
    - You need to use an Arduino which is compatible with the CC3000--check the product page (http://adafru.it/1469) for the latest compatibility information.
- CC3000 shield (http://adafru.it/1491) or breakout (http://adafru.it/1469)
- Infrared light sensor (http://adafru.it/157) and Infrared LED (http://adafru.it/387)
    - Make sure the IR sensor and LED are matched to the same light wavelength (usually 940nm). The IR sensor should be one tuned to detect 38khz signals.
- LED (http://adafru.it/777) (any color) as an indicator of the IR sensor receiving signals.
- Two 1/4 watt resistors:
    - R1: ~100 to ~1000 ohm resistor to limit the current to the IR LED.
    - R2: ~200 to ~1000 ohm resistor to limit the current to the indicator LED.
- Power source for the Arduino, such as a 9-volt battery (http://adafru.it/67) or wall wart (http://adafru.it/63).
- Solid core hookup wire (http://adafru.it/1311)
- Breadboard (http://adafru.it/64) or perf-board (http://adafru.it/571)
- Candy bowl or container you can mount the IR sensor and LED within.
- Hot glue or double stick tape to secure the sensor and LED inside the bowl.
- Soldering tools (http://adafru.it/136) to attach wire to the IR LED and sensor

Wiring

Connect your hardware as follows:



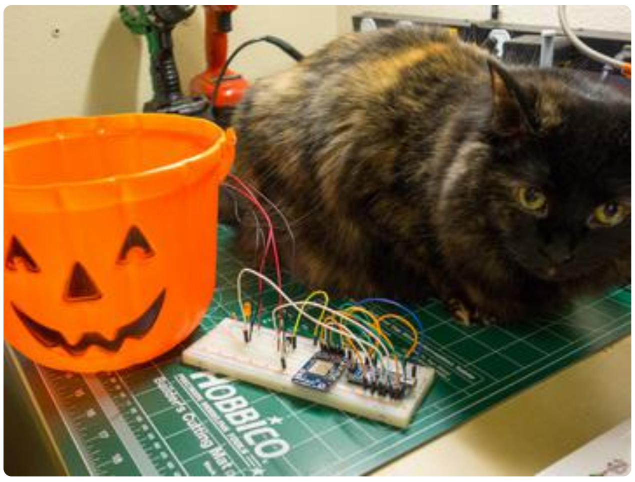
For the CC3000, connect it to the Arduino in the same way as this CC3000 tutorial ():

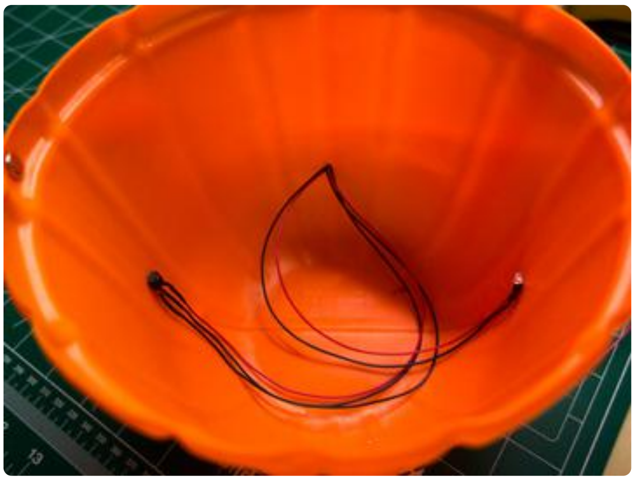
- Arduino 5V to CC3000 VIN
- Arduino ground to CC3000 ground
- Arduino digital pin 13 to CC3000 CLK
- Arduino digital pin 12 to CC3000 MISO
- Arduino digital pin 11 to CC3000 MOSI
- Arduino digital pin 10 to CC3000 CS
- Arduino digital pin 5 to CC3000 VBEN
- Arduino digital pin 3 to CC3000 IRQ

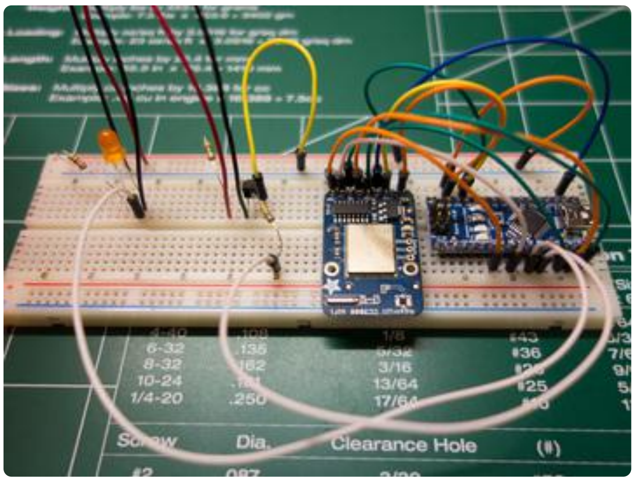
For the IR LED, solder long wires (long enough to reach from the bowl to the breadboard) to each leg--don't forget which leg is positive (longer) and negative! Connect the positive wire from the IR LED through to 5V power from the Arduino. Connect the negative wire from the IR LED through the R1 ~100 ohm resistor to Arduino digital pin 8.

For the IR sensor, again solder long wires to each leg. Connect the Vs leg to 5V power, the ground leg to ground, and the output leg to both Arduino digital pin 7 and the negative leg of the indicator LED. Check the datasheet () if you're unsure which pin is which on the sensor. Finally, hook up the positive leg of the indicator LED through the R2 ~220 ohm resistor to 5V power.

As a quick test, with the Arduino powered on you should be able to aim a remote control at the IR sensor and press buttons to see the indicator LED flash.

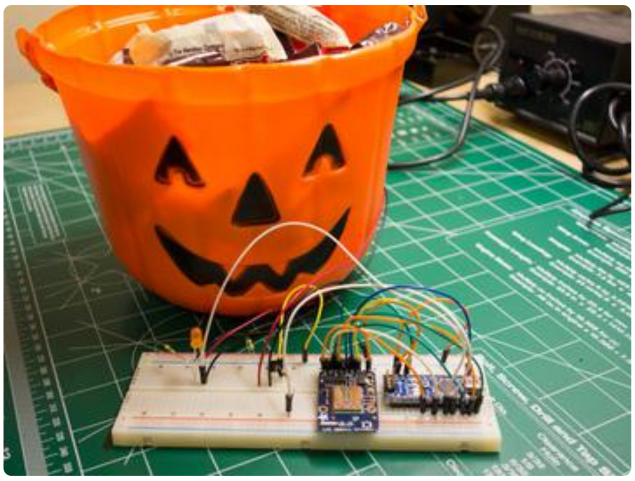






To finish the hardware, attach the IR sensor and IR LED to opposite sides that are facing each other inside the bowl. To the left you can see how I hot glued my sensor and LED about 1/3 of the way up the side of the bowl. You can also see the electronics that I built using an Arduino Nano, and a transistor to control current through the IR LED--the transistor isn't really necessary so it can be ignored.

If your bowl is big enough, you might be able to build a false floor with cardboard to hide the electronics neatly inside the bowl. Otherwise, run the wires outside the bowl and hide the electronics nearby.



Continue on to learn about the software used in this project.

Software

Once the hardware is setup, you can move on to running the software for the candy bowl monitor.

First make sure you have the latest version of the Adafruit CC3000 library installed. This project uses a server class which was recently added to the library so you need to update it to the latest version even if you have the library already installed. Check the CC3000 tutorial for information on installing the latest version of the CC3000 library ().

Next download the two sketches for this project with the link below:

Download Arduino Sketches

Unzip the archive and you'll find two Arduino sketches:

- Candybowl_Server
- Candybowl_Server_MDNS

Load the first sketch, Candybowl_Server, in the Arduino IDE and modify the defines at the top of the program. You can change which pins are hooked up to the CC3000, IR sensor, and IR LED. If you built the hardware on the previous page you shouldn't have to change any of the default values.

Also update the WLAN_SSID, WLAN_PASS, and WLAN_SECURITY defines to the appropriate values for your wireless network. See the CC3000 guide () for more information on configuring the CC3000--both these candy bowl sketches use the same configuration.

Compile the sketch and load it on your Arduino. Open the serial monitor at 115200 baud and watch for status from the CC3000 initialization to appear. After a few seconds you should see some details about the network and the message "Listening for connections..." printed. For example when connected to my network I see:

```
Hello, CC3000!

Initializing...
Started AP/SSID scan

Connecting to tdicola...Waiting to connect...Connected!
Request DHCP

IP Addr: 192.168.1.135
Netmask: 255.255.255.0
Gateway: 192.168.1.1
DHCPsrv: 192.168.1.1
DNSserv: 192.168.1.1
Listening for connections...
```

Make sure you see the listening for connections message before moving forward. If you see errors or have problems, check the CC3000 tutorial () to make sure you have everything hooked up and configured correctly.

Usage

To connect to the candy bowl server, open a telnet session on port 23 to the CC3000's IP address (you can see the IP address printed in the serial monitor, in my example it's 192.168.1.135).

On Mac OSX or Linux, open a terminal session and type:

```
telnet (CC3000 IP address)
```

For example if your CC3000 is on the IP address 192.168.1.135 you would type:

```
telnet 192.168.1.135
```

On Windows you'll need to install and use a telnet client. PuTTy () is a good, free graphical client to use.

Once connected to the server you can type a question mark character and press enter to have the server respond with the current candy bowl status. For example here's what you should see if the bowl is empty and queried, then filled with candy, and queried again:

```
> telnet 192.168.1.135
Trying 192.168.1.135...
Connected to 192.168.1.135.
Escape character is '^]'.
?
Candy bowl status: LOW
?
Candy bowl status: FULL
```

Note: Because this project uses light to detect if the bowl is empty or full, you might have problems if the candy inside the bowl is clear or see-through. Try to use candy that's dark and blocks light.

To close the telnet connection, on Mac OSX or Linux press ctrl-] to bring up a telnet command prompt and type quit. For PuTTY, just close the terminal window.

Connect to the server any time and query it for the current state of the candy bowl!

Code

If you are curious how the server works, look at the loop() function in the sketch. You can see the code below:

```
void loop(void)
{
  // Handle a connected client.
  Adafruit_CC3000_ClientRef client = candyServer.available();
  if (client) {
     // Check if there is data available to read.
     if (client.available() &gt; 0) {
       uint8_t ch = client.read();
       // Respond to a candy bowl status query.
       if (ch == '?') {
         client.fastrprint("Candy bowl status: ");
         if (isBowlFull()) {
           client.fastrprintln("FULL");
         }
         else {
           client.fastrprintln("LOW");
         }
       }
     }
  }
}
```

The Adafruit CC3000 library exposes a server interface that is very similar to the Arduino Ethernet library's server class (). If you have existing code that uses the Ethernet library server class, you should be able to port it to the CC3000 without many changes. The only difference is that the CC3000 library returns an 'Adafruit_CC3000_ClientRef' instance instead of an 'Adafruit_CC3000_Client' instance from the server's available() function. However you can use the client ref instance just like a client class and send or receive data to the connected client.

Continue on to learn about an enhancement to the server using multicast DNS.

Multicast DNS

One enhancement to the candy bowl monitor server is to use multicast DNS () to simplify connecting to the server. With MDNS the server can be assigned a local address like 'candybowl.local' which is used instead of the IP address to connect to the server. This means you don't need to know the IP address of the server (which can change each time the server connects to your network) and instead only need to know the fixed local address/name of the server.

To use MDNS with the candy bowl monitor, first download and install the CC3000 MDNS Arduino library (). A direct download link is below:

Download CC3000 MDNS Library

Unzip the library into a folder in your Arduino sketchbook/libraries directory. See the guide on Arduino libraries () if you aren't sure exactly what to do. Make sure to restart the Arduino IDE after installing the library.

Next you will want to make sure you have MDNS (or ZeroConf/Bonjour) support available to your operating system.

- For Mac OSX support should be provided automatically by Bonjour.
- For Linux, make sure Avahi () is installed. For recent versions of Ubuntu this should already be installed.
- For Windows, make sure Bonjour for windows () is installed.

Now load the 'Candybowl_Server_MDNS' sketch that was downloaded earlier. Configure this sketch in the same way as the first one, compile it, and load it on your Arduino.

Once the serial monitor displays the 'Listening for connections...' message, telnet to the 'candybowl.local' address. For example using a terminal session type:

```
telnet candybowl.local
```

After a short period of time you should see the telnet server find the IP address of the server and connect to it. You can send the same question mark command as the first server example to check the candy bowl status.

If you want to change the MDNS address of the server, just modify the call to the mdns.begin() function line in the setup() function of the sketch. By default the call looks like:

```
mdns.begin("candybowl", cc3000, 3600)
```

This means the server will listen for the 'candybowl.local' address (by convention all MDNS addresses end in '.local'). Try changing the value of the first parameter, recompiling, and uploading the sketch to have the server respond to a new address.

Also for reference the last parameter to the call specifies the time to live for the DNS record in seconds. In this case the sketch is setting a TTL of 3600 seconds, or one hour. This means your computer will cache the mapping of 'candybowl.local' to the server IP address for up to an hour. If you frequently disconnect and reconnect the server so the IP address changes, you might want to drop this down to a lower value like a few minutes. If you don't specify this third parameter, a default TTL of 1 hour will be used.

Feel free to use this simple MDNS library in your own server projects!

Continue on wrap up and look at future work you can do with this project.

Future Work

This project is a great example of how to build a server that can remotely monitor a sensor over WiFi with an Arduino and the CC3000. There are lots of interesting ways you can extend this project, such as:

- Use a strain gauge () to measure how much weight is in the candy bowl and calculate exactly how many pieces of candy are inside.
- Continuously pulse the IR LED and sensor to count how often a hand reaches into the candy bowl.
- Have the CC3000 send tweets () when the candy bowl is full or empty.
- Decorate the bowl with LEDs wired to a free digital pin and add a command to remotely blink or flash the LEDs from the telnet server.

- Make a simple web server to display the status of the candy bowl in a web browser. See the Arduino Ethernet library WebServer example () for the basic code you can make work with the CC3000 library.

Have fun with the WiFi candy bowl monitor project!