Smart Cocktail Shaker

Created by Tony DiCola

https://learn.adafruit.com/smart-cocktail-shaker

Last updated on 2023-08-29 02:30:11 PM EDT

Table of Contents

Overview

You will be the life of the party with the Smart Cocktail Shaker!

The smart cocktail shaker is a project to help you easily mix drinks using an Arduino, a load cell from a cheap kitchen scale, and an Android application. By measuring the weight of a cocktail shaker, an Arduino can send the amount of poured liquid to an Android application over a USB or bluetooth connection in real time. Making a drink is as easy as following the steps on screen--no more guessing or fumbling with measurements!

Hardware

You will need the following hardware for this project:

- Arduino Uno (http://adafru.it/50), Nano, or Mega (http://adafru.it/191).
- Android device running at least Android 3.1 (Honeycomb MR1), and with support for either bluetooth or USB host mode. A Nexus 7 tablet () is perfect for this project and supports both bluetooth and USB host mode.
    - Note that even if your Android device has a USB port it still might not support USB host mode! Unfortunately there's no single list of Android devices with or without USB host mode support so you might need to search the web for your specific device.

- Bluefruit EZ-link breakout (http://adafru.it/1588) or shield (http://adafru.it/1628) if using bluetooth to communicate with the Android device.
- USB on-the-go cable (http://adafru.it/1099) if using USB host mode to communicate with the Android device. Note that a USB OTG cable is not the same as a normal USB cable!
- Digital kitchen scale that you're willing to take apart and scavenge for the load cell. Try to find a scale that measures a few pounds with less than a gram accuracy. I found this 1000 gram scale from Harbor Freight tools () is perfect for this project--it's inexpensive, easy to take apart, and has all the wires from the load cell marked.
    - As an alternative, you can look at using a scale with a serial output (). These scales are typically more expensive, but will be easier to use since they're already calibrated and ready to use out of the box.
- Texas Instruments INA125 () instrument amplifier to amplify the small signal from the load cell. You can use other amplifiers, but this one is nice because it comes in a breadboard friendly DIP package and has a precision voltage reference to excite the load cell.
- Two 10k 25 turn trim potentiometers (). You can use other trim pots but pick ones which have a fairly high number of turns so you can precisely adjust the offset and gain of the instrument amplifier.
- 0.1 micro-farad ceramic capacitor to decouple V+ for the instrument amplifier.
- 1 micro-farad capacitor to connect the Bluefruit DTR line to the Arduino reset line for programming the Arduino over bluetooth ().
- Terminal block (http://adafru.it/677) to connect the tiny load cell wires to larger breadboard-friendly wires.
- Power supply in the 7 to 12 volt range, such as this 9 volt supply (http://adafru.it/63). Grab a barrel jack to alligator clip adapter (http://adafru.it/1328) to easily connect to the power supply too.
- Hookup wires (http://adafru.it/153) to connect components on the breadboard.
- Breadboard (http://adafru.it/239) to hold all the components.
- Precision screwdriver (http://adafru.it/424) to adjust the trim potentiometers.
- Multimeter (http://adafru.it/850) to measure the voltage from the instrument amplifier during calibration. Any simple meter should work.
- Soldering iron (http://adafru.it/180) to desolder load cell wires from the scale circuit board.
- A second scale or object with a known weight to use for scale calibration.

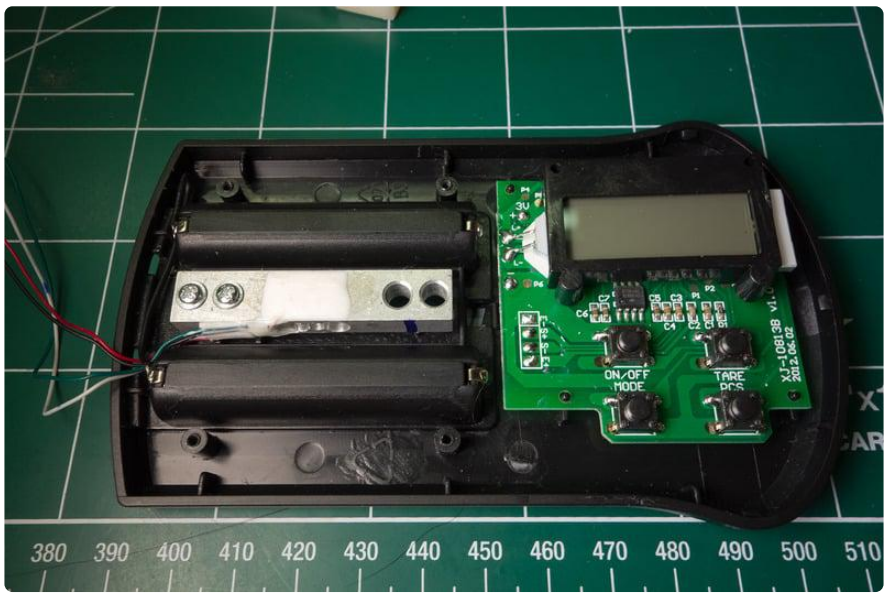
Kitchen Scale Tear Down

You will need to take apart the digital scale to gain access to the load cell. The exact disassembly method will vary depending on the scale, but in general you're looking

for the metal bar that sits directly below the measurement platform of the scale. This metal bar, or load cell, will have strain gauges glued to its sides and should have four wires coming out of it.

To disassemble the Harbor Freight scale I used in this project, start by removing two screws from inside the battery compartment and gently pull the platform from the top of the scale. Next remove the four screws revealed underneath the platform and pry the top of the plastic case off the scale to expose the circuit board. You should see thin red, black, white, and green wires going from the circuit board to the load cell. Remove the hot glue blob strain relief from where these wires attach to the circuit board (dab rubbing alcohol with a q-tip around the hot glue to cleanly remove it). Take note of which color wire goes to which label on the circuit board (there should be an E-, S+, S-, and E+ label). Desolder the four load cell wires from the circuit board with a soldering iron. Finally desolder the two circuit board power wires from the battery holder and remove the scale's circuit board.

You can see a picture of the disassembled scale below. The load cell is the metal bar to the left of the circuit board. Also note the 4 labeled connections for the load cell wires on the left of the circuit board: E-, S+, S-, E+



The load cell works by measuring the very small movement, or deflection, of the metal bar when weight is applied. Strain gauges glued to the metal bar change their resistance based on the bar's deflection. By putting these strain gauges in a special configuration, a Wheatstone bridge (), it's possible to measure the change in resistance as a change in voltage. The voltage from the load cell's bridge can be read by an analog input on an Arduino to determine the weight applied to the load cell.

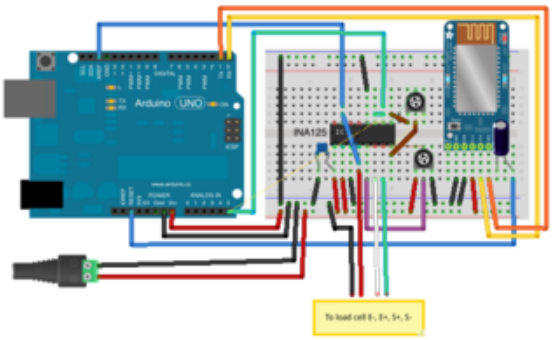
You can actually use a multimeter to see the change in voltage from the load cell as weight is applied. Connect the E+ wire to a battery or power supply positive terminal (anything 3-12 volts should work), the E- wire to the negative terminal, the S+ wire to the positive multimeter probe, and the S- wire to the negative multimeter probe. Set the multimeter to measure voltage in the millivolt range (if it's not auto-ranging). Apply weight or press on the load cell and watch what happens to the measured voltage. You should see the voltage increase as more weight is applied (if you see the voltage decrease, swap which probe is connected to the S+ and S- wires).

One problem with the load cell is that the voltage output is very small and difficult for an Arduino to directly read. You can see at maximum weight the load cell will only output a few millivolts. To make this small signal readable by an Arduino it will need to be amplified and buffered. An instrument amplifier, such as the Texas Instruments INA125 (), is a device which can amplify the signal from a load cell bridge and make it readable by the analog to digital converter in the Arduino.

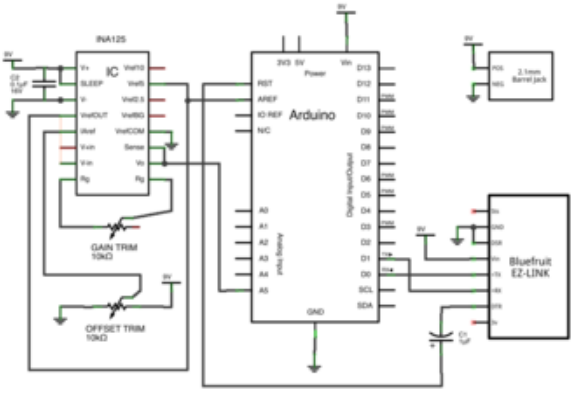
If you're curious for more information about load cells, check out these resources:

- Measuring Strain With Strain Gauges () by National Instruments
- Bridge Measurement Systems () by Texas Instruments

Assembly



Assemble the hardware as shown in the diagram and schematic to the left.



The Bluefruit EZ-link is connected to the Arduino in the same way as the Bluefruit tutorial suggests (). If you aren't using the Bluefruit, omit the wires and connections for it from your hardware.

For the INA125, if you're unsure consult the data sheet () for the name and meaning of each pin. You should connect the pins for this chip as follows:

- V+ to the power supply positive terminal, and V- to the power supply ground. A 0.1 micro-farad ceramic capacitor should be placed from V+ to ground to decouple power supply noise.
- SLEEP to V+, as directed by the data sheet.
- Vref OUT to Vref 5, the Arduino analog reference pin, and the load cell E+ wire. This connection will be the precision 5 volt reference from the INA125.
- IAref to the wiper, or middle pin, of a 10k trim potentiometer. One end of the potentiometer should be connected to ground, and the other end to the power supply positive (the exact choice of ends doesn't matter). This signal will offset the output of the amplifier into a stable range (more about this on the next page).
- V+in to the load cell S+ wire (white wire on the scale I used).
- V-in to the load cell S- wire (green wire on the scale I used).

- The Rg pins 8 and 9 at the end of the chip should be connected to the wiper and one end of the other 10k trim potentiometer. Changing the resistance across these Rg pins with the potentiometer will change the gain of the amplifier.
- Vo to Sense and an Arduino analog input such as A5. This is the amplified load cell signal that will be read by the Arduino.
- Vref COM to ground.

Finally connect the load cell E- wire to ground.

Continue on to learn how to calibrate the instrument amplifier and load cell.

Calibration

Instrument Amplifier Calibration

Once the hardware is assembled you will need to calibrate the output offset and gain of the instrument amplifier. The output offset is a small voltage that will be applied to the output of the amplifier and is necessary to push the amplifier into a stable, linear range. Because the amplifier is being used with a single positive voltage supply (as opposed to a dual supply with positive and negative voltages), the output of the amplifier is limited to a low value of only about 0.3 V above ground. With no weight applied to the load cell the output is likely below this 0.3 V limit so the behavior of the amplifier can be unstable and inaccurate. By applying voltage to the IAref pin on the amplifier (by using a trim potentiometer as a voltage divider) the output of the amplifier will be adjusted up above 0.3 V into a stable range.

To calibrate the output offset, connect your multimeter positive probe to the Vo output pin of the amplifier and the negative probe to ground. Apply power to the hardware and observe the voltage on the output pin when no weight is applied to the scale. Slowly turn the trim potentiometer connected to the IAref pin until the voltage on the output pin is around 0.5 volts.

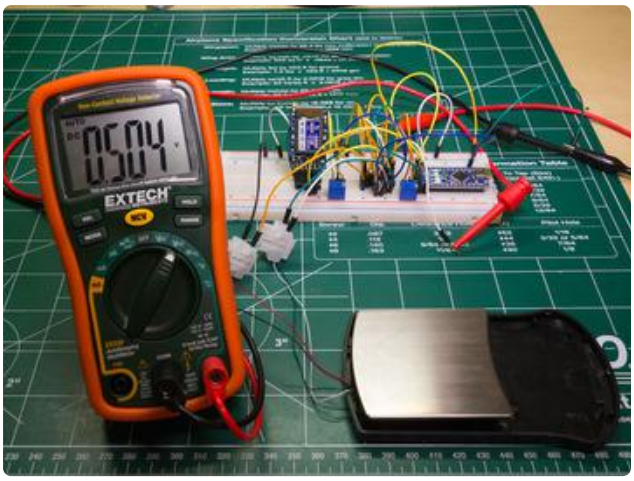
Next you must calibrate the gain of the amplifier. The easiest way to do this is to place a maximum amount of weight on the scale and adjust the gain trim potentometer until the output of the amplifier is 5 volts (the maximum voltage for the Arduino analog input). A good maximum weight for this project is a cocktail shaker or heavy glass filled completely with water (since this is likely the largest amount of weight to expect in a mixed drink). With the multimeter still connected to Vo and ground, place a glass that is full of water on the scale. Adjust the trim potentiometer connected to the Rg pins until the voltage on the output pin is around 5 volts.

Note: If you see the voltage drop on the output when weight is applied to the scale,

swap the V+in and V-in wires (load cell signal wires) and try the calibration again.

After adjusting the gain, double check the output offset with no weight applied is still around 0.5 volts. Readjust both the output offset and gain as necessary to get them near the 0.5 and 5 volt values with no weight and maximum weight.

Below is a summary of the calibration on my hardware:



Adjust the output offset trim potentiometer with no weight on the scale until the output voltage is around 0.5 volts.

Adjust the gain trim potentiometer with a full glass of water on the scale until the output voltage is around 5 volts.

Check that when some weight is removed from the scale, the voltage drops to a value between 0.5 and 5 volts. The voltage on the output of the amplifier will be proportional to the weight on the scale.

Arduino Setup & Calibration

In this step you'll calibrate the Arduino sketch for the project so it can accurately measure weight applied to the scale. You will need another scale to measure the weight of an object like a glass or shaker to use in the calibration. If you don't have a second scale, look for an object with a known weight like a pile of coins ().

Download the software for this project at the following link and unzip the archive to a folder.

Download Software

Load the ScaleCalibrate sketch in Arduino and adjust the ADC_PIN #define at the top of the sketch if you're using an analog input other the default A5. Apply power to the hardware and upload the sketch to the Arduino.

Note: If you have a Bluefruit EZ-link hooked up but are uploading over USB, disconnect the power and ground of the EZ-link temporarily so it does not interfere with the upload of the sketch. Also on an Arduino Nano I found it was necessary to remove the Arduino's Vin power pin while communicating over USB (this might not be necessary for other Arduinos).

Open the Arduino serial monitor and change the baud rate to 9600. You should see a message like the following:

Scale Calibration Sketch

Type OK and press enter to start.

If you don't see the message, try resetting the Arduino.

Follow the prompts on the serial monitor and type OK and enter to start.

The calibration will tell you to remove all weight from the scale and type OK and enter to continue.

Next the calibration will tell you to place something on the scale and enter its weight in grams. For example if I had an object that was 120.5 grams I would place it on the scale and type 120.5 and enter to continue.

Finally the calibration will output two values you need to save for use later. For example I saw these values with my calibration:

Calibration finished! Write down the following calibration values:
ZERO_OFFSET = 103.00000
GRAMS_PER_MEASUREMENT = 1.00061

If you type OK and press enter the sketch will enter a loop where it prints the weight (in grams) of whatever is on the scale. Try placing items of various known weights on the scale to confirm it is reasonably accurate. Based on the Arduino's 10-bit ADC and a full glass of water as maximum weight, the scale will only have around 1 gram resolution.

Now load the SmartCocktailShakerSketch in Arduino and adjust the ZERO_OFFSET and GRAMS_PER_MEASUREMENT #define values with the values you found from calibration. Also adjust the ADC_PIN value for your hardware if necessary. Upload the sketch to your Arduino, open the serial monitor at 9600 baud, and confirm if you type ? (a single question mark) and press enter the hardware responds with the weight on the scale in grams.

At this point the hardware is setup and ready to use with the Android application. Continue on to learn how to use the Android application with a USB connection.

USB Communication

To use the project with a USB connection on your Android device, make sure your Android device is running at least Android version 3.1 and supports USB host mode. You will also need a USB on-the-go or OTG cable to connect to your Arduino. The USB OTG cable tells the Android device to enter USB host mode and supply power to the Arduino. A quick check to see if you device supports USB host mode would be to connect a USB keyboard to the USB OTG cable and connect it to your Android device. If you can use the keyboard on your device, chances are good it will work with USB host mode to communicate to an Arduino.

To load the software on your Android device you will need to enable loading applications from unknown sources by following the instructions at this link ().

Download the SmartShaker.apk file to your Android device from the following link:

Download SmartShaker.apk

Your device should prompt you if you want to install the application. Agree to install the application and it will be loaded on your Android device.

With power applied to your project hardware, connect the USB OTG cable to a USB mini or micro cable which is connected to your Arduino. Run the Smart Shaker application (it will have a green star as an icon) and follow the instructions to make a drink. Watch the second half of the video from the overview () for an example of using the project with a USB connection.

Note: As mentioned in the previous step, if you also have a Bluefruit EZ-link attached to your hardware make sure to disconnect it before using the USB connection. Also on an Arduino Nano it might be necessary to disconnect the Vin power line to the Arduino so it is powered from USB instead of the power supply.

Android USB Development

If you're interested in building your own Android applications which communicate with Arduno devices over the USB connection you can look at the source code for the Smart Shaker application included in the software download. A full overview of Android development is beyond the scope of this guide, but if you want to learn about Android development check out these resources:

- Google's Android Developer webpage ()
- Google's Android Training tutorials ()
- Android tutorials from vogella.com ()

Also be aware the source for this project was built using the Android Studio () development environment, and is not directly importable into the ADT + Eclipse environment ().

For the USB communication, this project uses the usb-serial-for-android library () which is a great wrapper around USB to serial communication for Arduino and common FTDI chips. I've included the source to this library in the application and made a few small changes to it:

- Added an InputStream and OutputStream implementation to better integrate the USB serial device with other Java classes.
- Added a function to enumerate all the USB devices which are supported by the library. The default device enumeration function in the library requires permission to all the attached devices which is not really feasible for use in a device selection list.

It will also be helpful to familiarize yourself with Android's USB development documentation (). Although the usb-serial-for-android library does most of the work for you, you will still need to do things like add permission for USB host mode () to your application manifest.

Bluetooth Communication

To use the project with a bluetooth connection you will need an Android 3.0 or greater device with bluetooth support. You will need to pair your Android device with your Bluefruit EZ-link before you run the Smart Shaker application. If you aren't sure how to pair with a bluetooth device, follow these instructions ().

Once your Bluefruit is paired with your Android device, make sure the hardware is powered up and run the Smart Shaker application that was downloaded in the previous step (). Watch the first half of the video from the overview () for an example of using the project with a bluetooth connection.

Android Bluetooth Development

If you want to use bluetooth to communicate with a Bluefruit EZ-link in your own project, here are some tips I found while developing this project:

- Limit your Arduino sketch Serial communication speed to 9600 baud. This step is very important, if you try other baud rates the Android device will not be able to communicate with the Bluefruit EZ-link. Communication issues with Android and Bluefruit are still being investigated, but for now stick with 9600 baud to prevent issues.
- Follow Android bluetooth best practices and perform all bluetooth communication outside the main UI thread. The Android bluetooth APIs are blocking and can take a few seconds to run so attempting to call them in an activity UI thread will hang your application. The smart shaker application uses a Timer () to run device communication on a separate thread (and a Handler () to send updates back to the UI thread). An AsyncTask () is another option for offloading bluetooth communication to a separate thread.

The Android bluetooth documentation () is a good resource for learning more about bluetooth development.

Future Work

This project is a great example of how to hack a kitchen scale into a tool for making drinks with an Android application. You can use the code from this project as an example of how to communicate between an Android device and an Arduino through a USB or bluetooth connection.

Some interesting ways you can extend the project are:

- Add your own drink recipes by modifying the drinks.json () file and rebuilding the application. Look at the existing drinks to see the schema for documenting drink preparation steps.
- Use a higher precision analog to digital converter, like this 16-bit ADC (http://adafru.it/1085), to measure much smaller weights from the load cell. Perhaps you can even build a smart measuring cup to simplify tasks in the kitchen like baking cookies!
- Add an audible warning to the application when nearing the target measurement for a preparation step.
- Try switching the application to use a SQLite database to store a huge number of drink recipes. Look at normalizing the grams_per_oz for each preparation step into a separate table based on each ingredient type.
- Use an Android device to add a great interface to your own Arduino project!