24/07/2020

Motor Phase Current Sensing

HANDOVER DOCUMENT – SUMMER INTERN PROJECT

PHIL BLECHER

Table of Contents

Set-up ***1***

Real Set-up ***2***

Current monitoring ***2***

Comparing Signal Conditioning Options ***4***

Next Steps for the Project ***5***

Handover Folder Explanation ***5***

Set-up

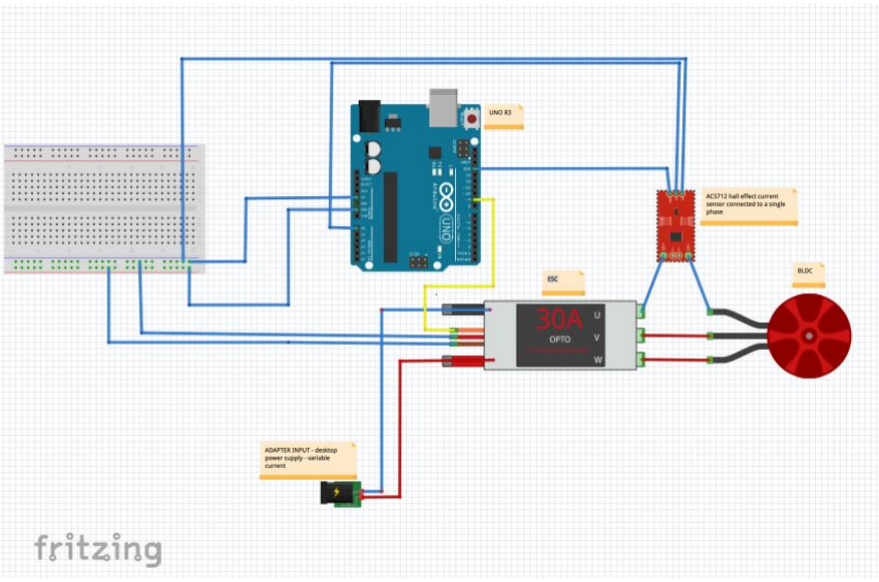

Figure 1 Test circuit diagram

Using the set up above, current can be monitored with either an Arduino sketch[1], showing the current in a serial monitor, or through the use of Processing and Arduino combined as an oscilloscope.

[1] All sketches have been sourced from online examples and open source software.

Real Set-up

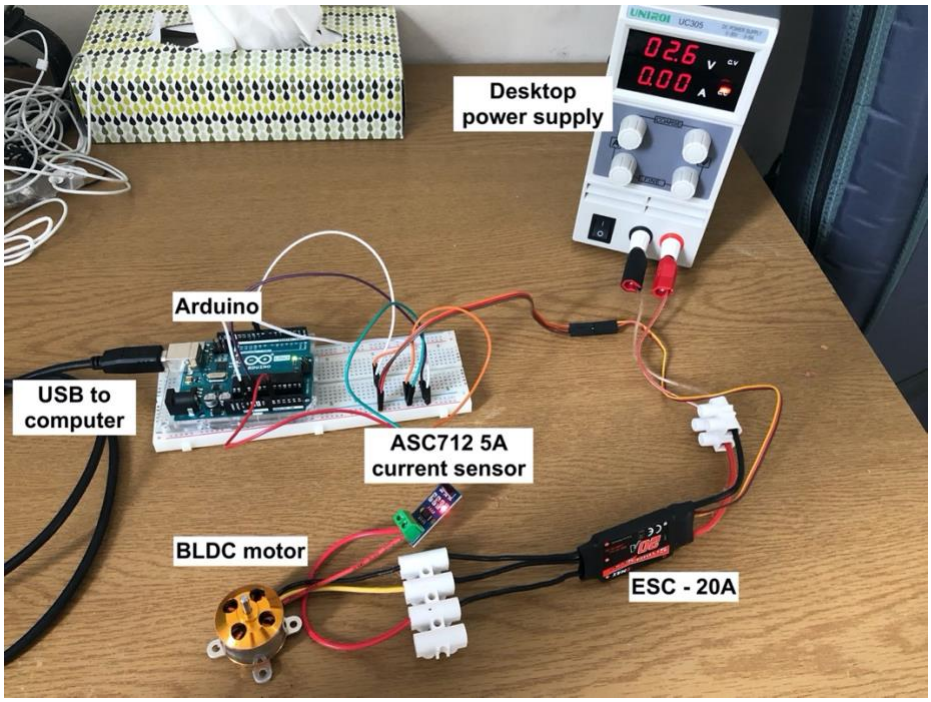

Figure 2 Real life set up

Current monitoring

Comparing the serial monitor trace, using only Arduino, and the oscilloscope trace using Processing, it is clear that through additional software the signal can be shown much more accurately.

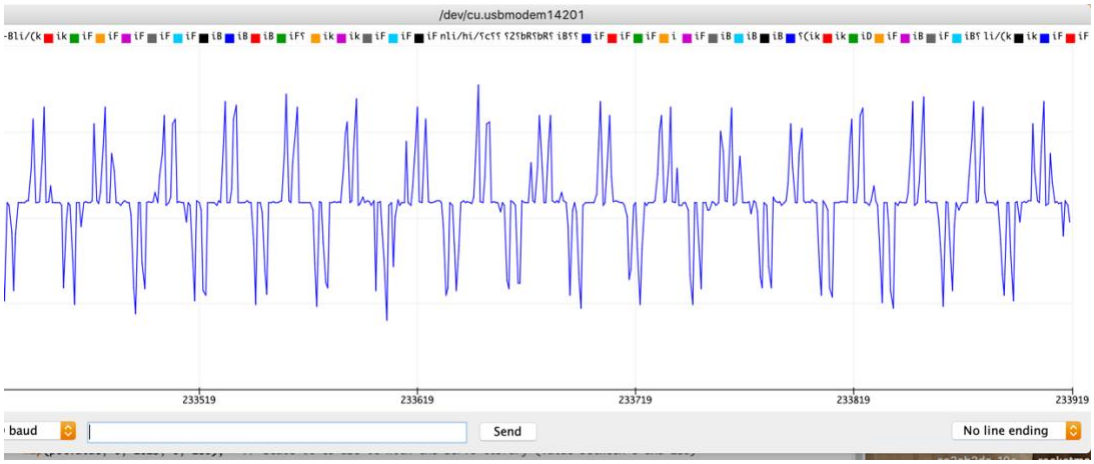

Figure 3 Arduino serial monitor

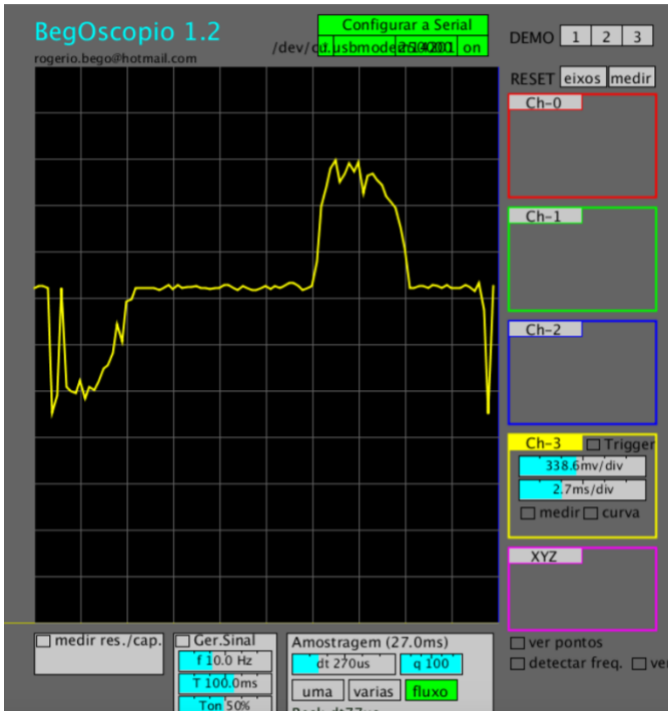

Figure 4 Processing oscilloscope sketch of the current.

I have altered an oscilloscope sketch[2] created in Processing and Arduino to also run the motor at a constant input power, the speed can be altered with a variable desktop power supply or by introducing a potentiometer into the circuit with feedback to the Arduino. This is explained in comments in the sketch[3].

For accurate current readings the voltage output **needs to be calibrated** from the sensor. The formula below[4] demonstrates the relationship for the ACS712 chip that is given on the data sheet. This relationship needs to be experimentally confirmed through constant load and known current testing.

$$V_{out} \approx a \cdot I_{in} + b$$

Using a basic averaging function[5] it is possible to extract a much smoother set of continuous data. Using the Filters library, it is possible to apply data averaging, digital high pass and low pass filters. The results of these data processing options are shown below. The relationship used to obtain these values is shown in the equation below. These values are sourced from the Instructables link[6] where a calibration has been carried out on the same chip.

$$V_{out} \approx 0.0405 I_{in} - 0.1129$$

This appears to bring the value of current closer to the expected value. Although, these values need to be confirmed.

```
//LOW PASS FILTER
 FilterOnePole lowpassFilter( LOWPASS, 1000 );


//HIGH PASS FILTER
FilterTwoPole highpassFilter( HIGHPASS, 10 );
```

Figure 5 Use of both a high pass and a low pass filter.

[2] https://github.com/rogeriobego/oscilloscope-arduino-processing/files/1109314/oscilloscope_arduino.zip
[3] See handover folder
[4] https://create.arduino.cc/projecthub/instrumentation-system/acs712-current-sensor-87b4a6
[5] https://www.instructables.com/id/Simplified-Arduino-AC-Current-Measurement-Using-AC/
[6] https://www.instructables.com/id/Simplified-Arduino-AC-Current-Measurement-Using-AC/

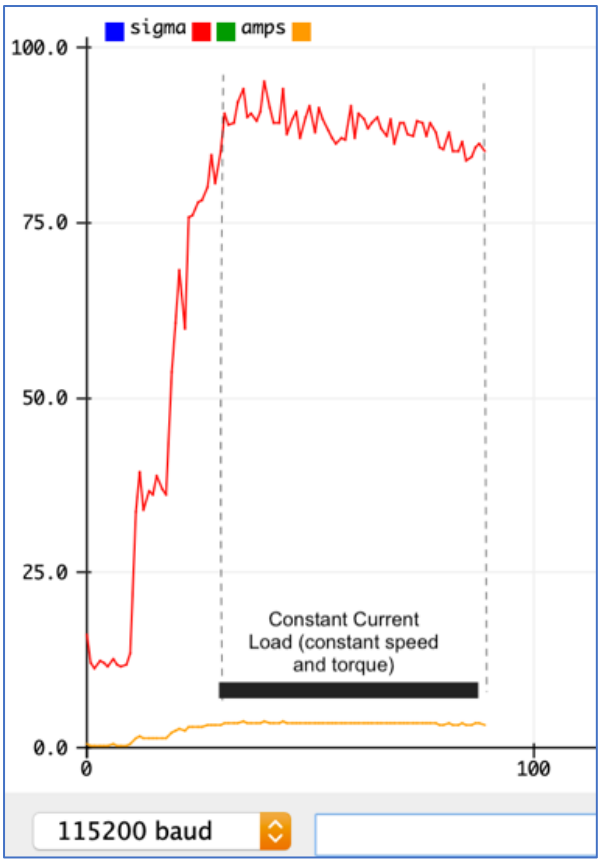

Figure 6 Filtered signal trace in Arduino serial monitor

As shown in figure 6, although uncalibrated, the Arduino sketch with running averaging can show a representative trace of the signal. In the figure, the sigma value represents the raw value the Arduino reads on the analogue input. This value is then converted to the amps value through a calculation of the relationship between the input and output as mentioned above.

To detect transient current changes, the length of averaging can be altered at the beginning of the sketch. Filters can be applied in order to remove noise and highlight steps in the current.

Comparing Signal Conditioning Options[7]

1. **Data Averaging**
    a. Averages the data in a given time frame.
    b. Fast and removes random noise
    c. Retains steps in signal features
2. **Low Pass Filter**
    a. Does not retain discontinuities as well as an averaging filter
    b. Removes random/noise spikes
    c. Causes slightly delayed step changes
3. **High Pass Filter**
    a. Same as low pass filter

[7] https://seeq.atlassian.net/wiki/spaces/KB/pages/512786844/Intro+to+Signal+Smoothing+Filters#IntrotoSignalSmoothingFilters-WhentoFilter

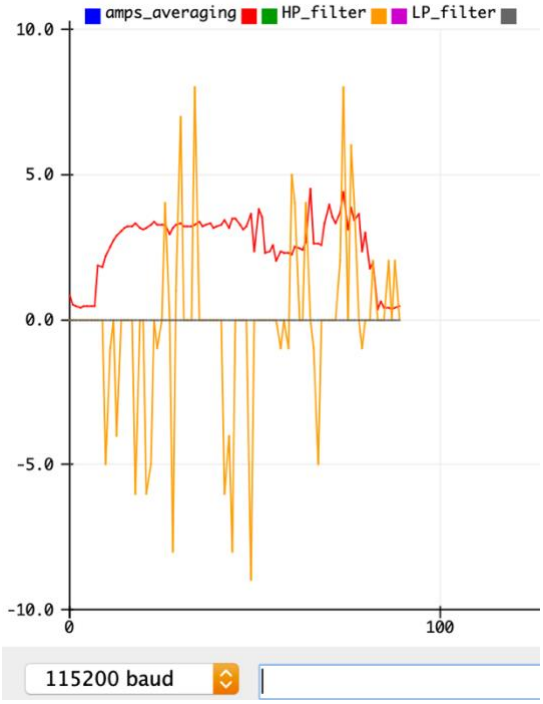

Figure 7 Serial plot of the motor phase current

- The signal that is passed through the running data average (Red) follows the current level the best with the closest to the true value
- The high pass filtered signal (Orange) does not show the current trace however does show rapid changes in the current accurately.
- The low pass filter does not show any signal after experimenting with different values of cut-off frequency.

Next Steps for the Project

- Calibrate sensor using a known constant current load and insert this into code.
- Create a way of detecting current spikes from the high pass filter trace/running average trace and create a trigger for these occasions.
- Compare analogue filtering to digital.

Handover Folder Explanation

In the handover folder there are separate folders for the Arduino programming that I used, the Fritzing circuit diagram and the images and videos taken throughout the project to demonstrate progress.

- Use the sketch in folder labelled “Code for Arduino motor control and serial monitor” to control the motor in the setup in figure 7 and take a reading of a single phase of the current using data averaging and a high pass filter as shown in figure 6.
- Download Processing[8] and use the files in folder “Code for Arduino and Processing” to create an oscilloscope and monitor the current.
- The Fritzing diagram can be used for reference when creating the circuit.

[8] https://processing.org