Indian Institute of Technology, Bombay

EE 344: Electronics Design Lab

Lock-in Amplifiers with Fluxgate sensors: Design and Implementation

Milestone III Report

Joshi Tanmay Ajay (200070027)
Neeraj Prabhu (200070049)
Siddhant Midha (200070078)

Last updated on December 7, 2023

1 Fluxgate

1.1 Description of test setup

Before starting we review the basic principle:

Principle: The flxuxgate sensor is a device which detects magnetic fields. This is done by applying a square wave voltage on the primary winding (wound around the magnetic core) to drive the magnetic core into saturation. Then, a secondary coil is wound around this apparatus. External (DC) fields are detected as voltage levels at a particular frequency in the secondary coil.

Let us elaborate on the principle, and discuss the waveforms at the output and what we expect so as to get a sense of the tests that we perform later. As written in the previous reports, we excite the primary coil with a DRIVE VOLTAGE of the following form (here, we exaggerate the $H \to L$ and $L \to H$ transitions for effect):

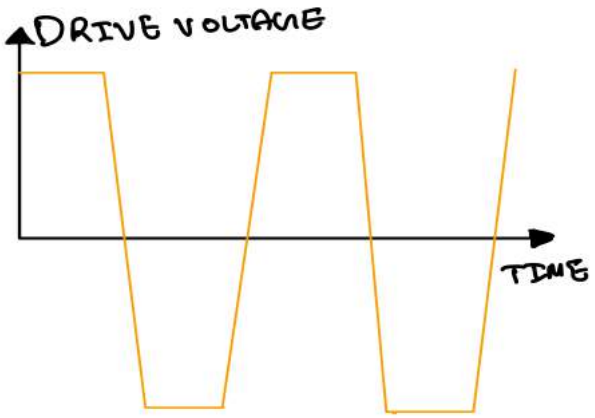


Figure 1: The DRIVE VOLTAGE

This causes the two half-core arms to be saturated, and the magnetic field waveforms for the two arms look as follows in the absence of any external field:

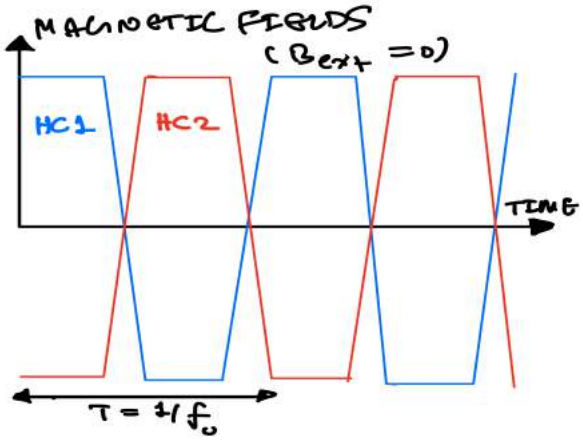


Figure 2: The FIELD WAVEFORMS with $B_{ext} = 0$

Now, when we turn on an external magnetic field, the waveforms become asymmetric. In one half cycle, the field by one of the arms is supported by the external field, and the other is demoted. This results in the waveforms being modified as follows.

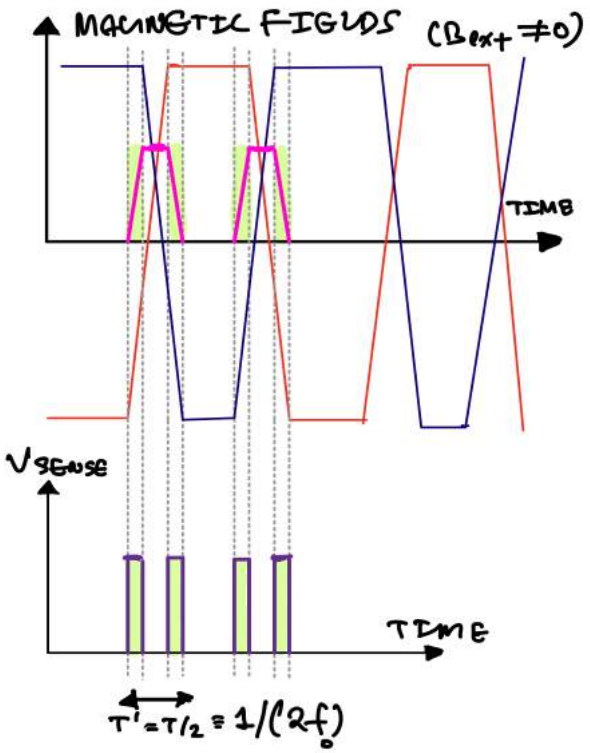


Figure 3: Assymetry in net field and induced voltage in sense coil

In the figure, the pink curve indicates the net field, which is non-zero during some times because of the external magnetic field. Now, we have indicated the regions wherein this net field is changing with a green-highlight. By basic electromagnetism, a voltage will be induced in the secondary coil terminals because of this change in magnetic field (flux). We have shown the corresponding excited voltage as V_{SENSE} in the figure. Thus, we expect such a waveform at the sensed signal. Moreover, it is clear that the frequency of the sensed voltage doubles. Thus, accounting for the noisy components etc., the information about the external field is contained in the $2f_0$ component.
We test the individual components of a fluxgate sequentially, all the way to the output. Let us give an overall block diagram of the circuit entailing the test-points which need to be verified.

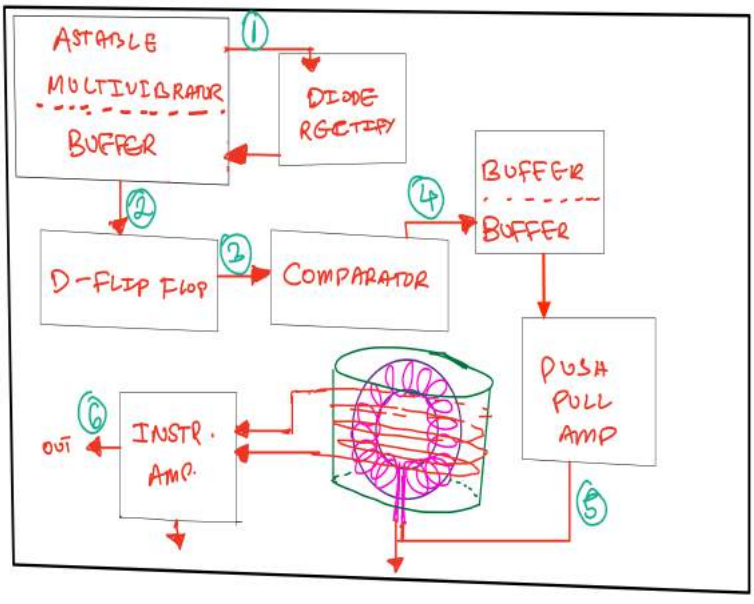


Figure 4: The Test Setup

We test the following points:

1. Raw output of astable-multivibrator
2. Astable waveform after being rectified (and buffered)
3. Waveform after frequency division
4. Waveform after comparison (so as to be sufficient for driving core into saturation)
5. Output of the push-pull amplifier ≡ waveform across the primary coil
6. Output of the instrumentation amplifier ≡ (Amplified and) sensed signal

1.2 Breadboard Test Results

1.2.1 Phase I

Here, we provide an overview of our preliminary breadboard rig-up.

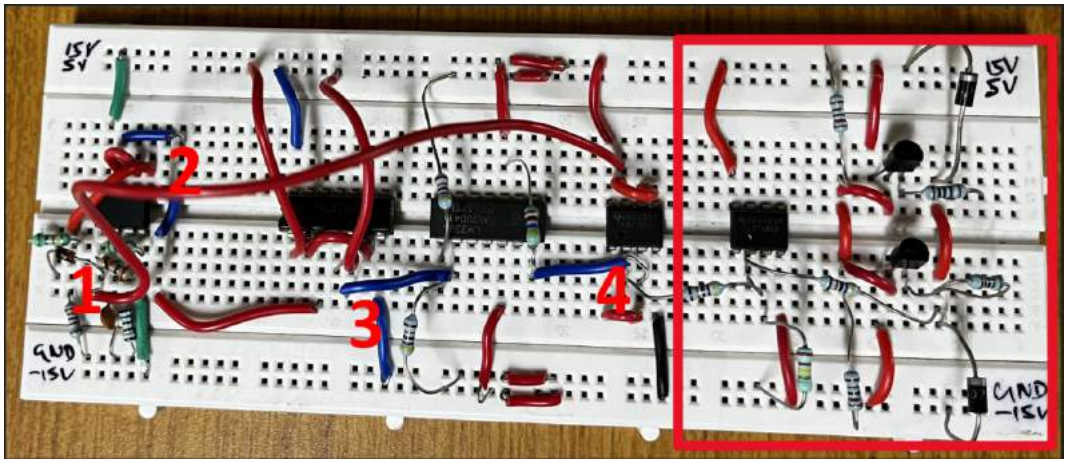
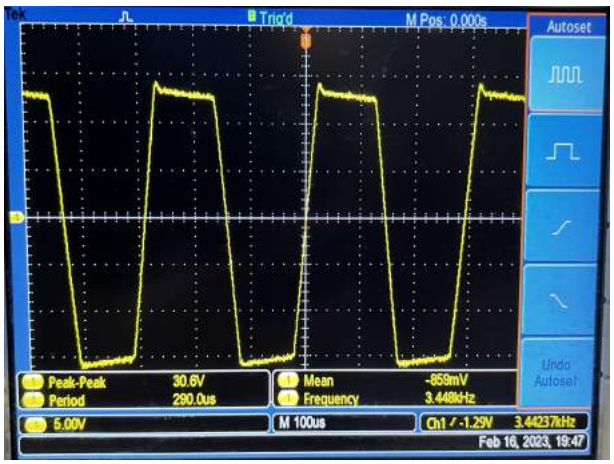
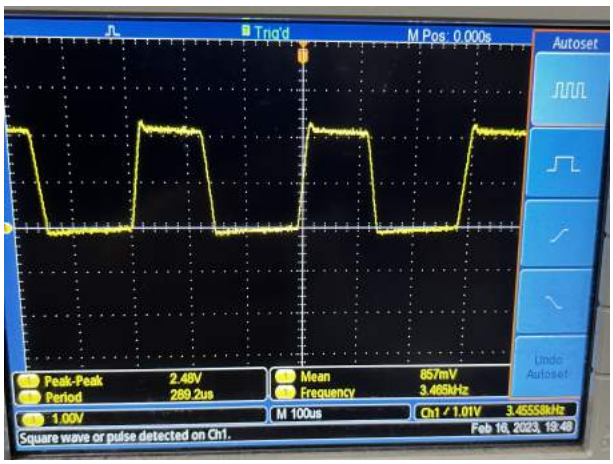


Figure 5: Full breadboard circuit: The numbers indicate points at which the voltages are shown below. The square box is the push-pull circuit, yet to be tested.

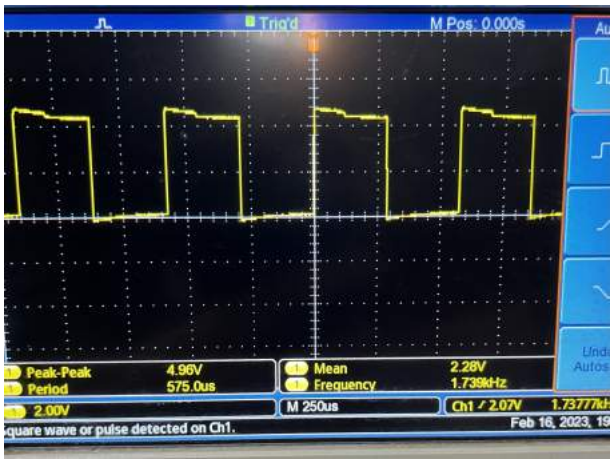
Here are the voltage outputs at important points in the circuit which have been tested yet.



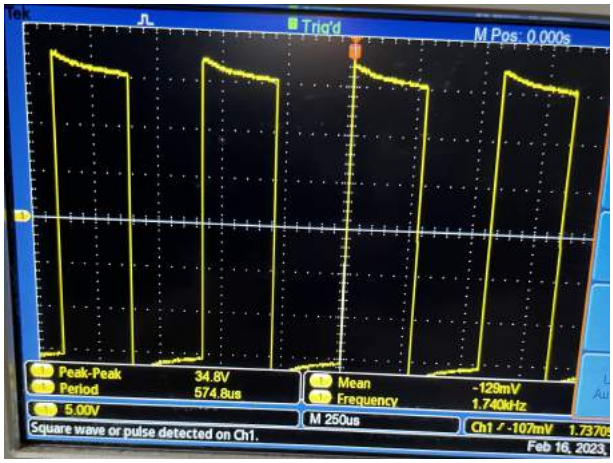
(a) Voltage probed at the output of the astable multivibrator (1)



(b) Voltage probed at the buffer after diode rectification (2)



(c) Voltage probed at the output of the flip-flop (3)(4)



(d) Voltage probed at the output of the comparator

Figure 6: Various voltages probed in the breadboard testing of the fluxgate initial circuit.

Moreover, here is a voltage measurement which clearly shows the frequency division performed by the flip flop.

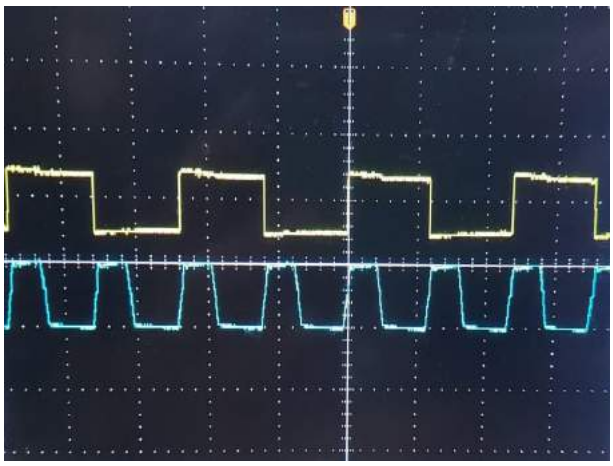


Figure 7: Flip flop output (yellow) compared with input (blue)

1.2.2 Phase II

Now, we test the push-pull amplifier as well: that is, whether it is capable of driving the ferrite core into saturation. This is a picture of the set up.

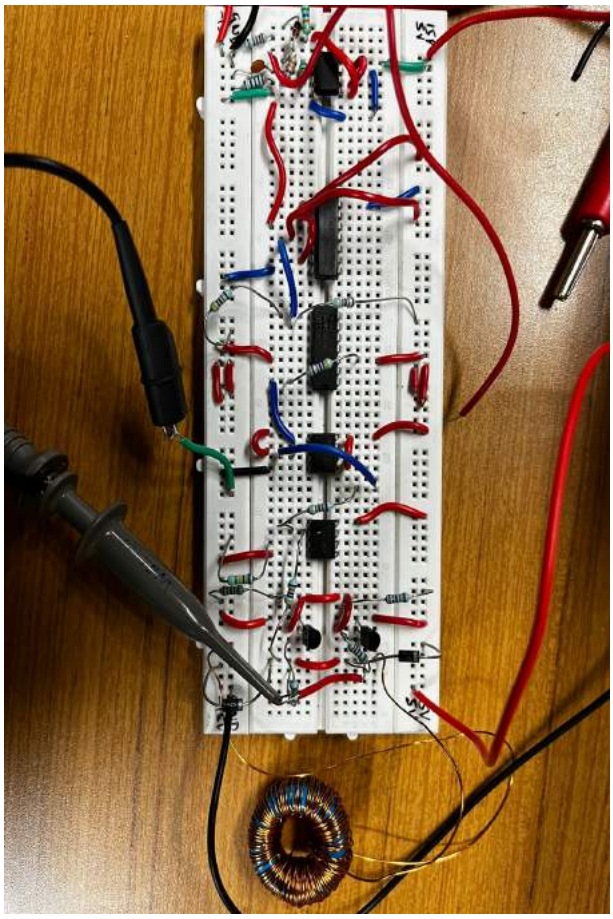


Figure 8: Circuit for push-pull testing and driving core into saturation

We use our internally generated frequency as input to the push-pull amplifier. We obtain the following.

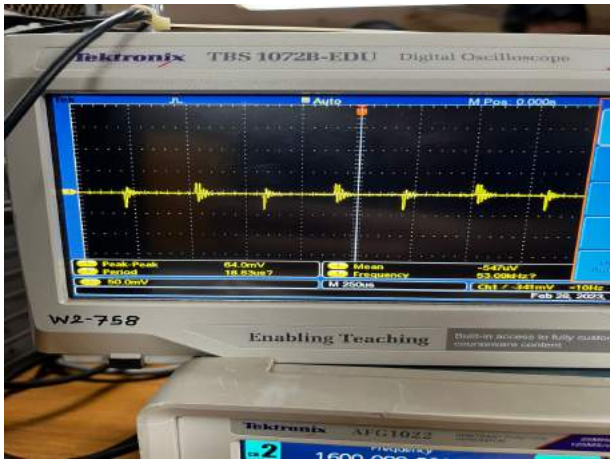
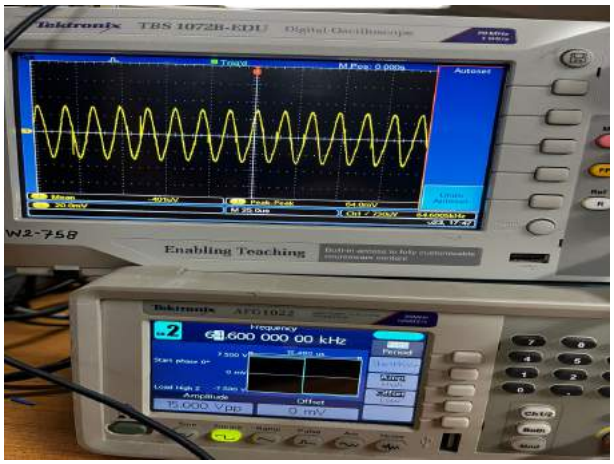


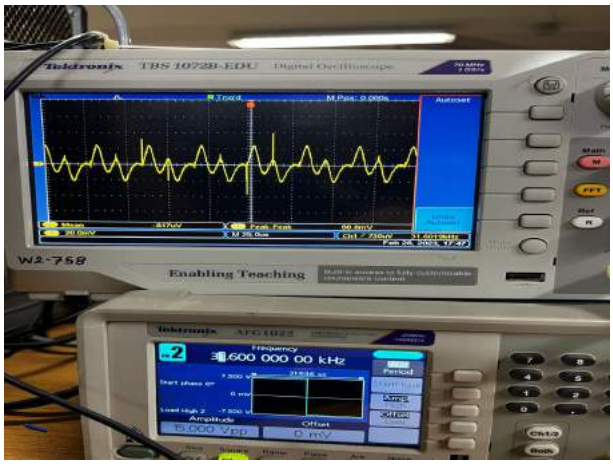
Figure 9: Voltage across the coil wound around the core when the push-pull circuit is excited using the internally generated signal

To prove that this is indeed saturation, we test this set-up against a signal externally

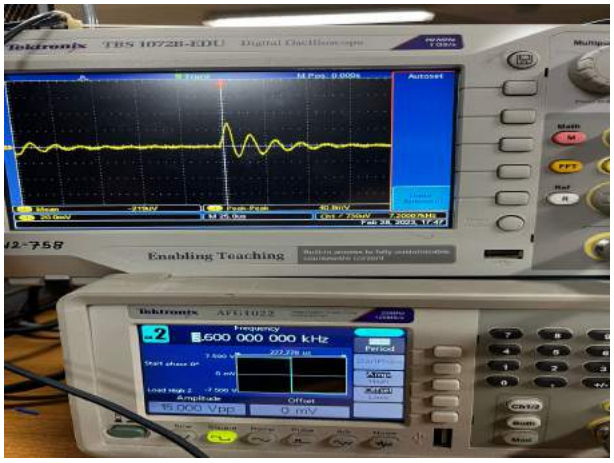
generated by an AFG. Here are the results at different frequency which show that the core goes into saturation at lower frequencies.



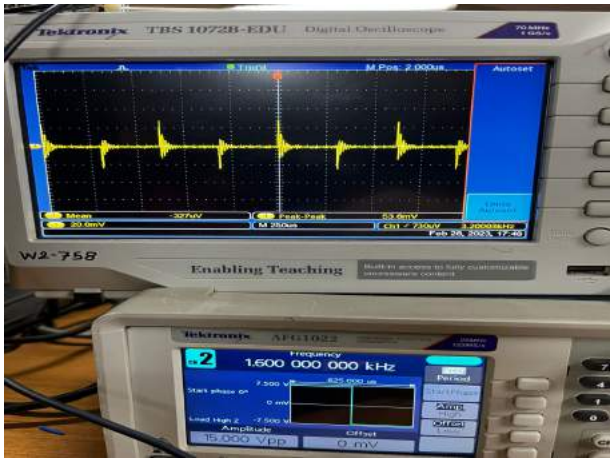
(a) At $f = 64.6KHz$



(b) At $f = 34.6KHz$



(c) At $f = 3.6KHz$



(d) At $f = 1.6KHz$ (*our operating frequency)*

Figure 10: Voltage across the coil wound around the core when the push-pull circuit is excited at multiple frequencies

An issue we see is that, due to a noisy breadboard and *many* stray $L - C$ combinations, we get a 'wiggly' curve at $f = 3.6KHz$ and if we zoom in at $f = 1.6KHz$. This needs to be solved by proper traces on a PCB.

1.2.3 Phase III

Now, we wind the secondary coil around the primary one (which is already wound on the toroid). We take the output from the secondary coil, which is due to the imbalance in the net (primary + external) flux caused by the external magnetic field. We pass this through an amplification stage. This is done using an instrumentation amplifier `INA128`. The gain of this amplifier is given as,

$$\text{Gain} = 1 + \frac{50K\Omega}{R_G}$$

where, R_G is the single resistor connected across two of it's terminals. Since we were getting a $\sim mV$ signal without the amplifier, we want a gain of the order of 100. Thus, we choose $R_G = 5K\Omega$, leading to a gain of 11.

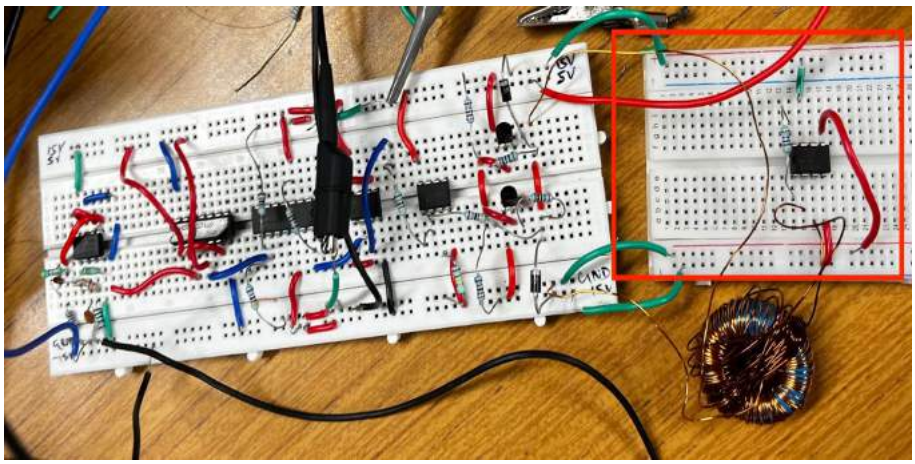


Figure 11: Breadboard circuit: Red box indicates the instrumentation amplifier for post-processing of the sensed signal before sending to the lock-in

We use a simple bar magnet to pedagogically illustrate the fluxgate outputs in the breadboard setup. We add the magnet as follows.

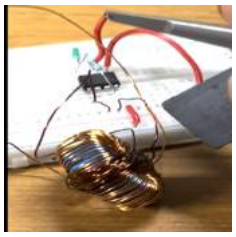
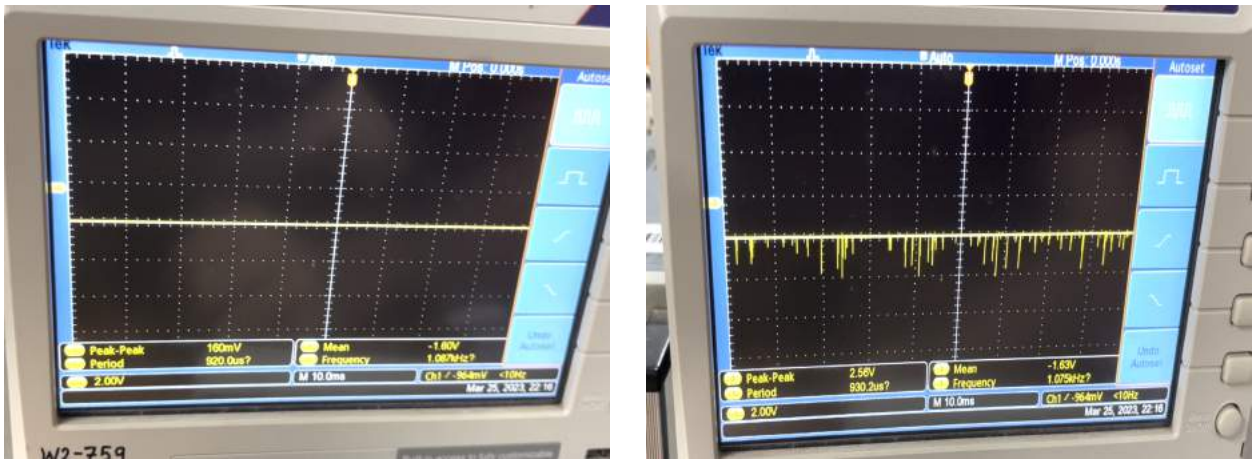


Figure 12: Shown: Bar magnet used for detection using the fluxgate sensor

Now, we show the output of the instrumentation amplifier (that is, the amplified sensed signal) at low and high magnetic fields.

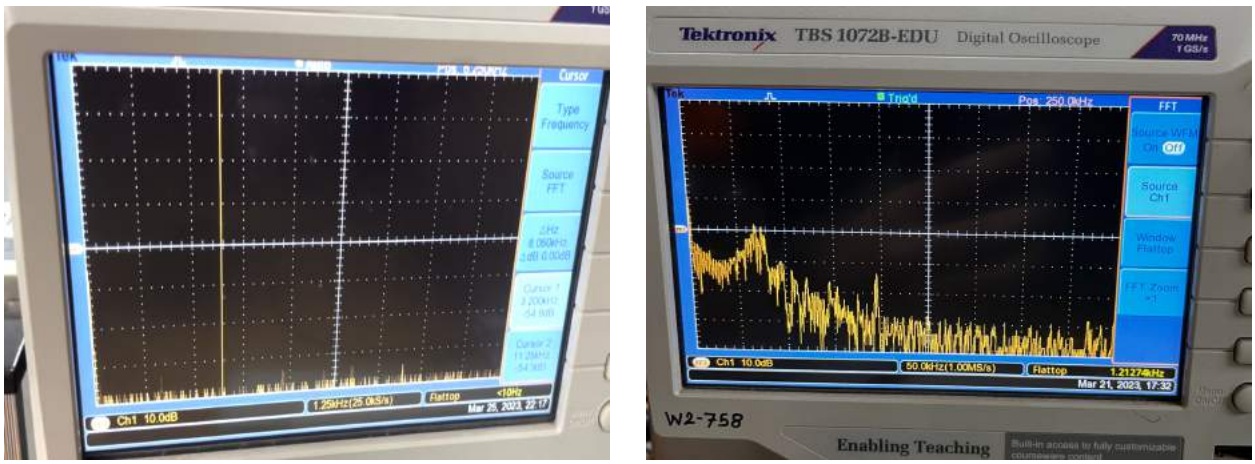


(a) Output of the instrumentation amplifier at low (none) magnetic field

(b) Output of the instrumentation amplifier at high magnetic field (using a bar magnet)

Figure 13: Outputs of instrumentation amplifier

We also show the FFT of the waveform above, showing a peak which contains information about the magnetic field. This can be extracted using the reference signal and this signal with a lock-in amplifier.



(a) FFT at low (none) magnetic field (b) FFT at high magnetic field (using a bar magnet)

Figure 14: FFT Outputs of the instrumentation amplifier

1.2.4 Observations and Test Results

Here, we first provide a percentage testing block-diagram:

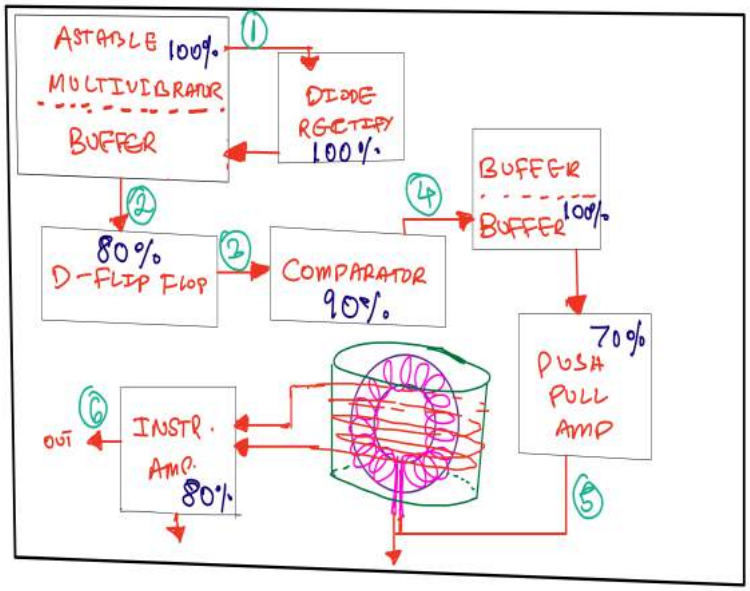


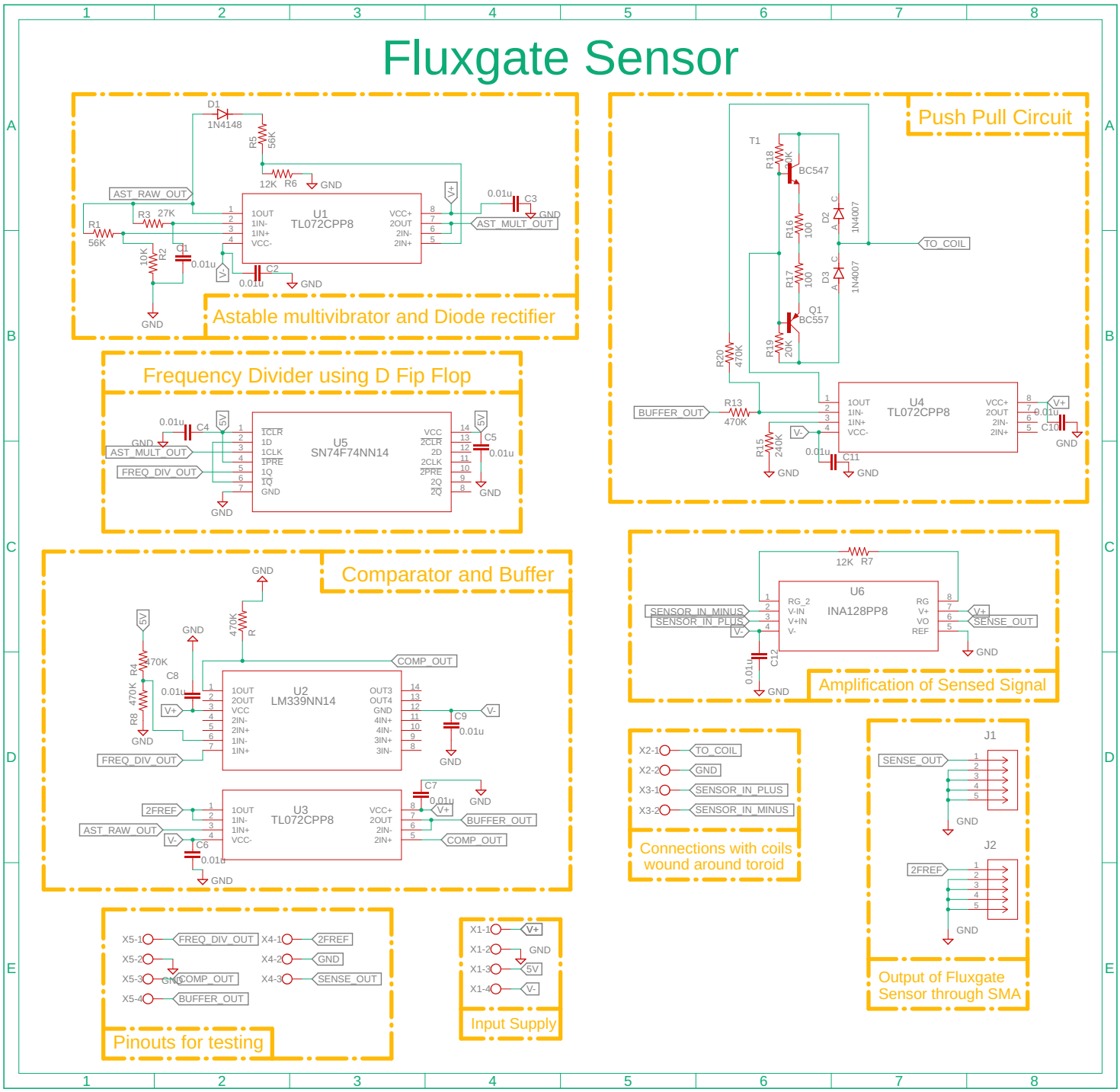
Figure 15: The Percentage Tested Block Diagram

Justification:

- Astable multivibrator and Buffer(s): Working precisely as expected, no major issues have happened yet and are not expected in the PCB either.
- Diode Rectifier: Simple circuit, working as expected.
- D-Flip Flop: An issue we observed earlier in testing was that sometime there was a need to set the `CLR-BAR` (BAR indicates `NOT`) pin of the D-Flip Flop IC to `LOW` so that the IC would function as expected. But, in the next rounds of testing, this was not needed. So, it is not clear whether that was an issue with a possibly noisy circuit, or a mechanism which is required. Hence, we assign a score of 80% as we will need to check this in the PCB as well.

- Comparator: While there are no major issues, there could be an issue with keeping the voltage symmetric and the voltage levels changing from the breadboard to the PCB, and possibly a need for change in resistor values around the comparator. While unlikely, this needs to be checked.
- Push-pull amplifier: We verified the voltages at the internal nodes only during the initial testing phase, and have not documented them further. The output voltage is as expected. Hence, to be transparent we assign it a score of 70%. We do not expect any failures.
- Instrumentation amplifier: A clearer idea of the value of GAIN (or equivaltently, the R_G) required before input to the lock-in will be obtained after integration only. Hence, we assign a score of 80%.

1.3 Circuit Schematic


Fluxgate Sensor
Astable multivibrator and Diode rectifier
Push Pull Circuit
Frequency Divider using D Fip Flop
Comparator and Buffer
Amplification of Sensed Signal
Connections with coils wound around toroid
Output of Fluxgate Sensor through SMA
Pinouts for testing
Input Supply
U1 TL072CPP8
U2 LM339NN14
U3 TL072CPP8
U4 TL072CPP8
U5 SN74F74NN14
U6 INA128PP8
AST_RAW_OUT
AST_MULT_OUT
FREQ_DIV_OUT
COMP_OUT
BUFFER_OUT
SENSE_OUT
SENSOR_IN_PLUS
SENSOR_IN_MINUS
TO_COIL
2FREF
GND
5V
V+
V-

1.4 PCB Layout

Before providing the PCB layout, we draw a sketch of the set-up. This is to clarify how the primary coil, which is wound around the toroid, and around which the secondary coil is wound is fixed to the PCB.

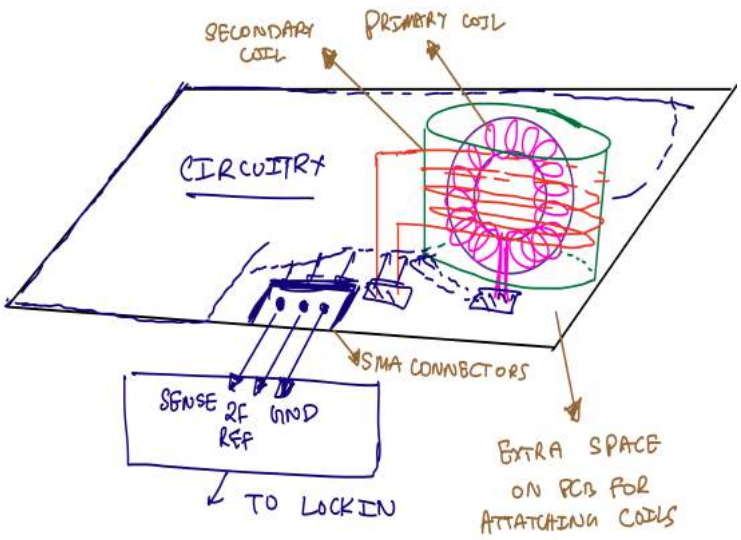
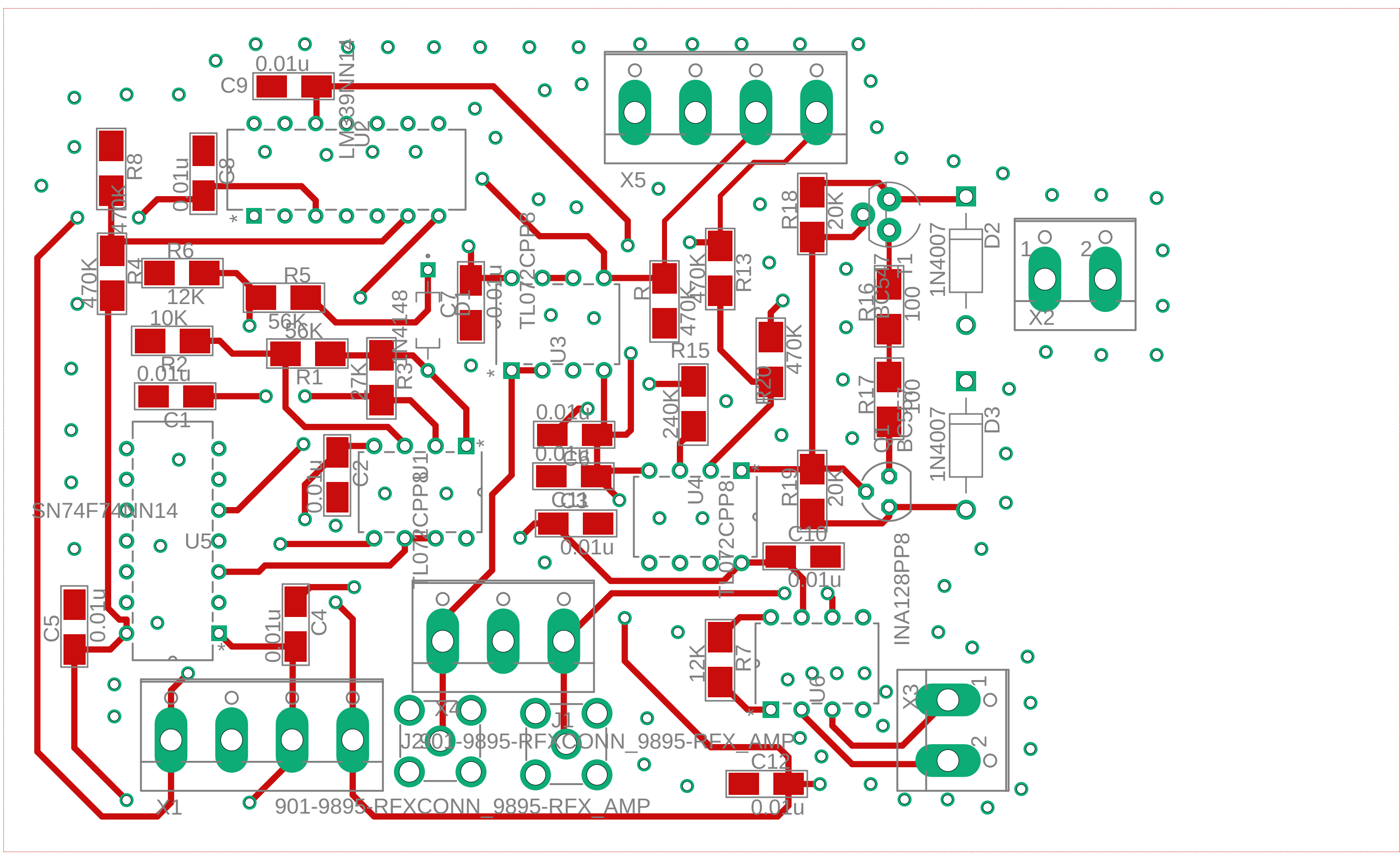
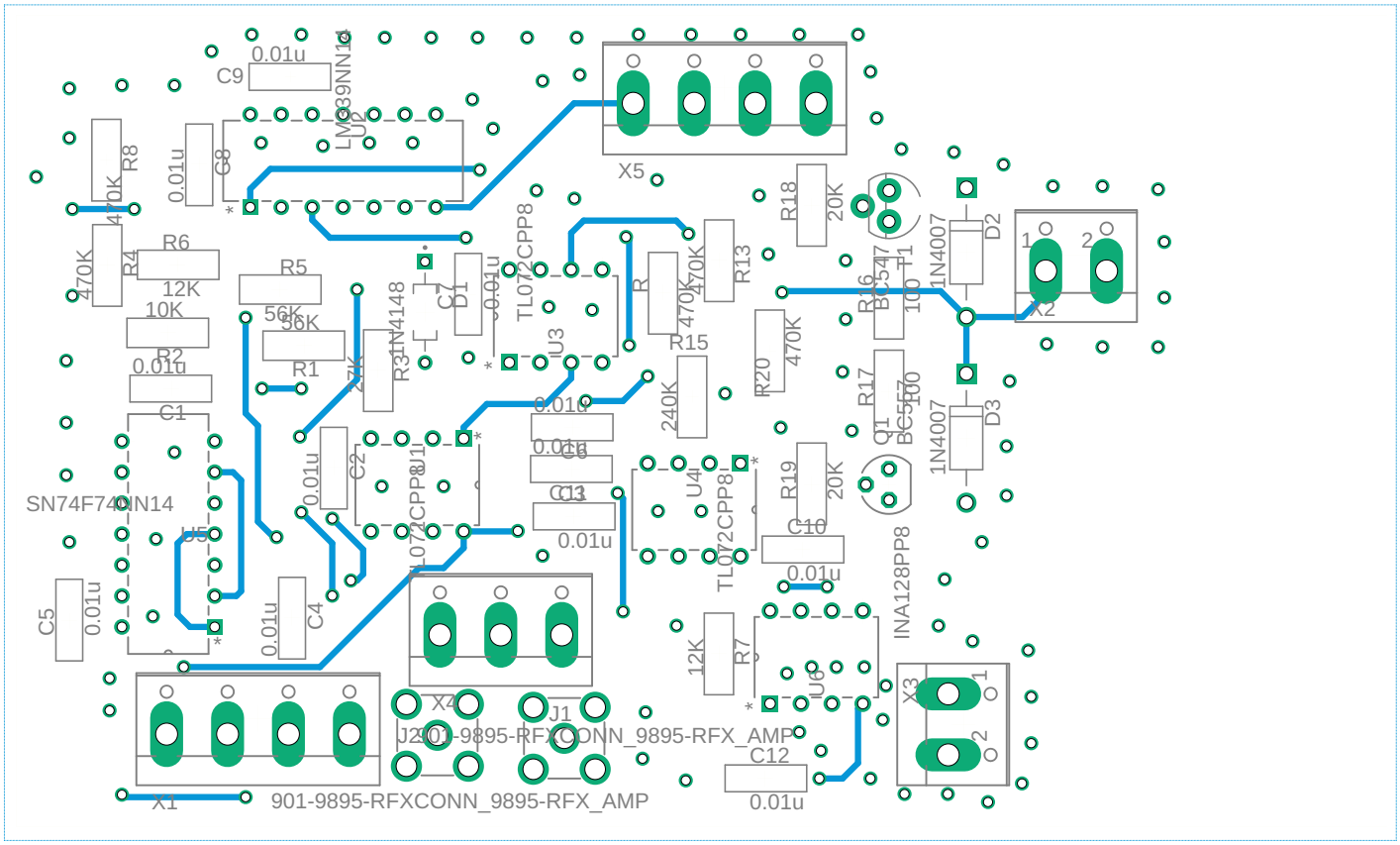


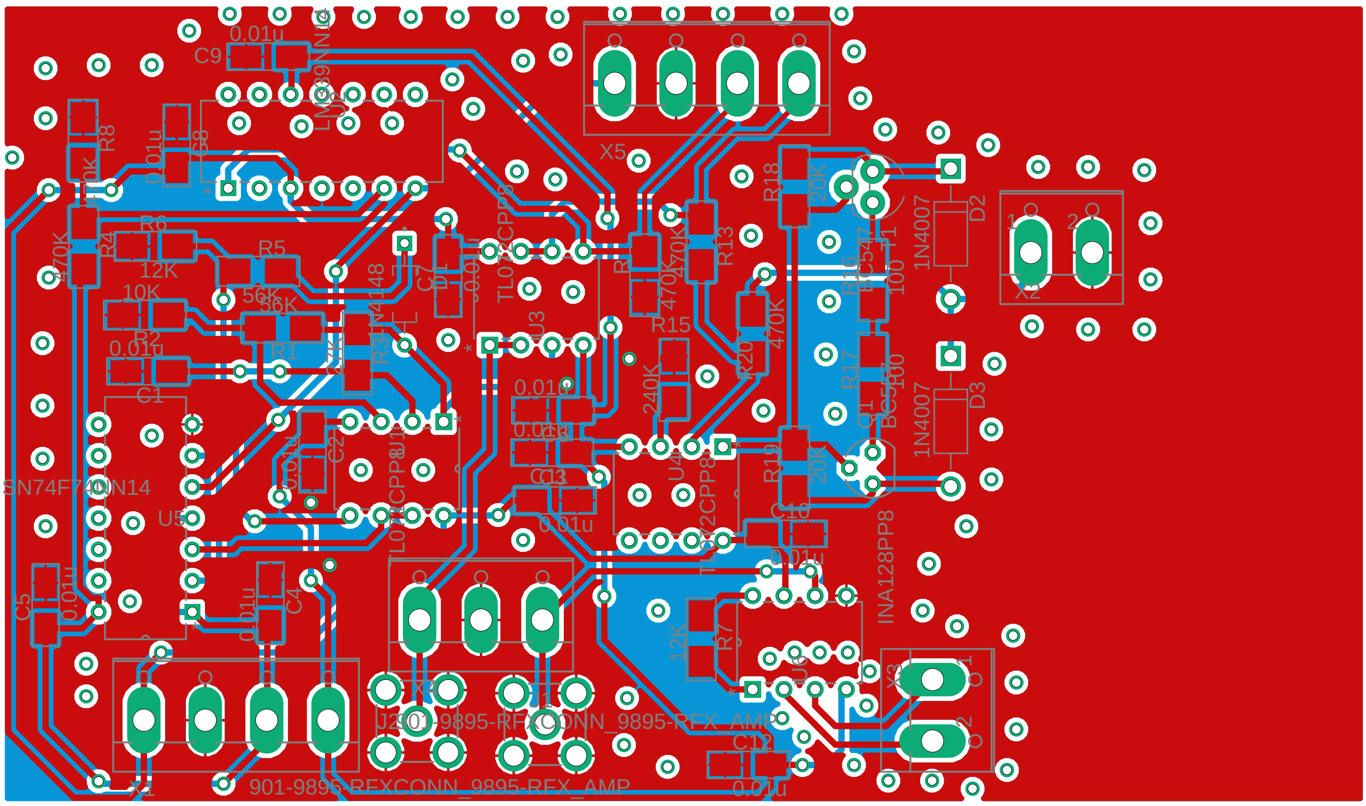
Figure 16: Sketch of the Coils and PCB setup integration

Note: as shown in the figure above, there is a container around which the secondary coil is wound. We will 3D print a container, in which the toroid + wound primary coil will be placed. Around this structure, we will wind the secondary coil. The CAD for this structure is in the next subsection.



26-03-2023 21:50 f=2.30 C:\Users\NEERAJ~1\AppData\Local\Temp\Neutron\ElectronFileOutput\14760\brd-8c7dd563-f47d-4458-9a90-e064fef2dbe3\Fluxgate_PCB v4.brd


0.01u
C9
LM339NN14
U6
R8
470K
0.01u
C8
X5
R18
20K
R6
R4
470K
12K
R5
56K
10K
R2
0.01u
C1
R1
2K
R3
1N4148
C7
D1
0.01u
TL072CPP8
U3
R
470K
R13
R15
240K
R16
BC547
Q1
100
F1
1N4007
D2
1
2
X2
R20
470K
R17
BC557
100
1N4007
D3
0.01u
C6
0.01u
C11
0.01u
SN74F74N14
U5
C2
0.01u
U1
TL072CPP8
U4
R19
20K
C10
0.01u
INA128PP8
C5
0.01u
C4
0.01u
12K
R7
U7
X4
J1
X3
1
2
X1
901-9895-RFXCONN_9895-RFX_AMP
C12
0.01u



26-03-2023 14:22 f=2.30 C:\Users\NEERAJ~1\AppData\Local\Temp\Neutron\ElectronFileOutput\21020\brd-594c86a9-133b-4a58-92e0-e99d37198d2f\Fluxgate_PCB v1.brd

1.5 CAD Layout

We designed a box-like structure to contain the toroid, which will be screwed to the PCB board. Further, we will wind the secondary coil around this box. A 3D view is as follows:

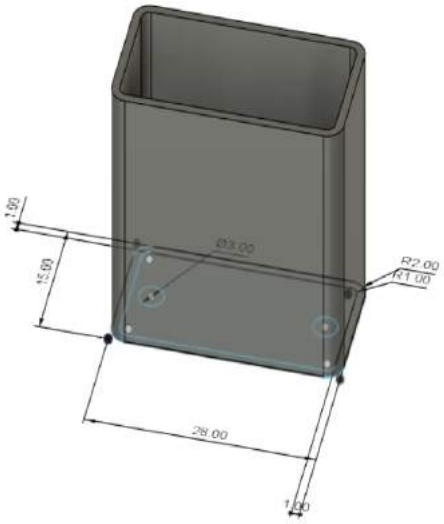
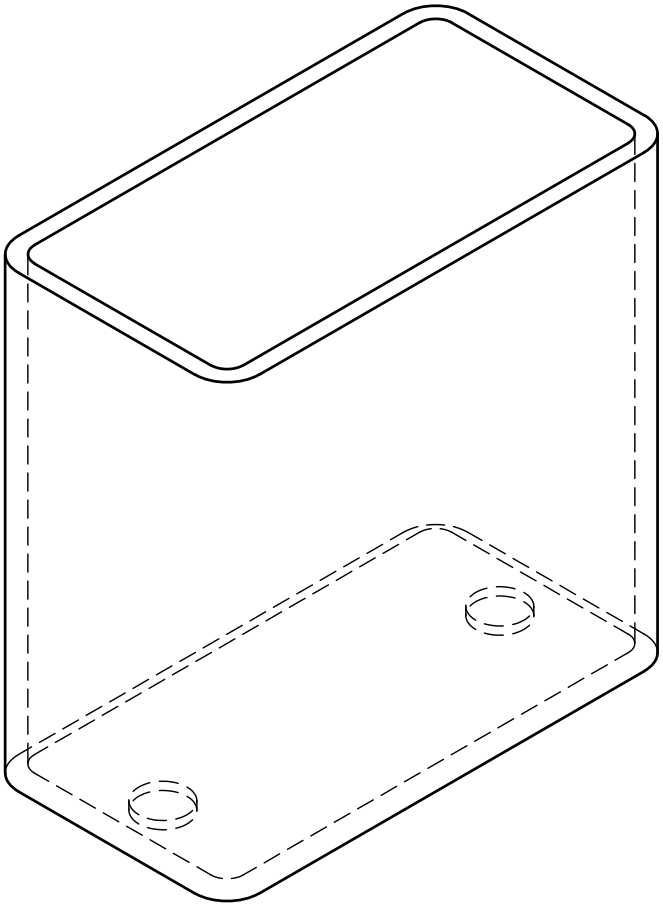
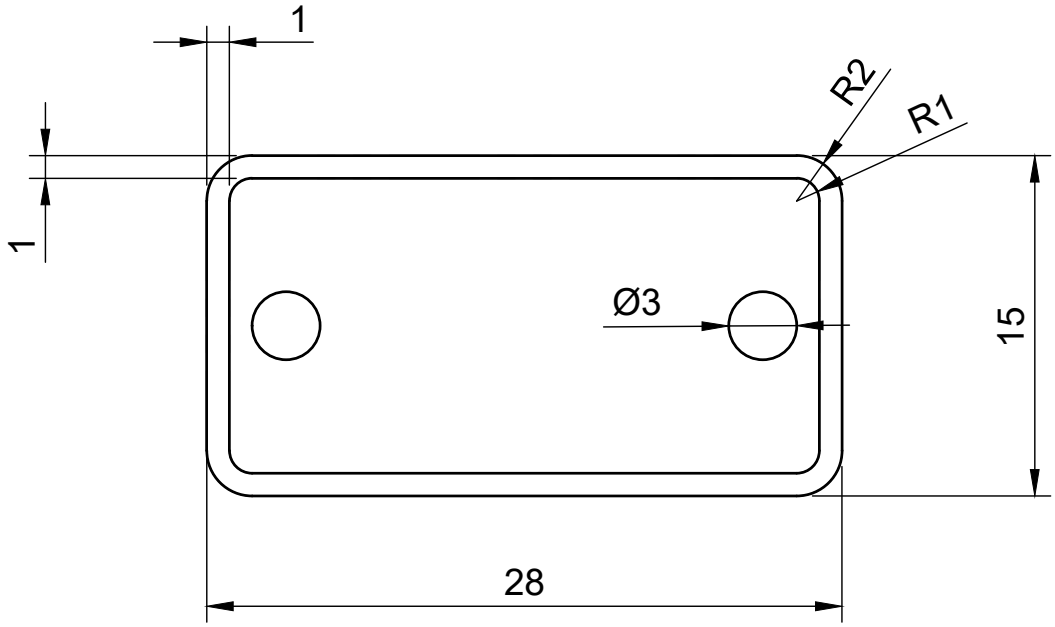


Figure 17: 3D Layout of the Box

Further, we provide the exact schematic of the CAD design in the next page.





Dept.	Technical reference	Created by Siddhant Midha 26-03-2023	Approved by	
		Document type	Document status	
		Title Toroid_Case	DWG No.	
			Rev.	Date of issue / Sheet 1/1

1.6 Next Steps

The remaining tasks in the fluxgate are as follows:

- The main task right now is getting the PCB printed and printing the CAD design. The CAD will not take long, but we expect the PCB to take 2 days to get printed. Meanwhile, we will focus on the lock-in amplifiers.
- After the PCB is printed, we will do testing of individual components and fix the toroid onto the PCB.
- Then, an end-to-end testing (without lock-in) will be done with necessary desbugging.
- Attach the SMA pins and the SMA cables, and start the interfacing with the lock-in amplifiers.
- Another round of debugging, as interfacing issues can arise.
- Design (not yet, after PCB testing) and print the overall package box for the fluxgate and package it.

A rough day-wise breakdown is:

Day	Task
Mar 27-28	3-D Print the CAD design, send the PCB for printing (focus on lock-in)
Mar 29-31	Receive the PCB: Do component testing through test-points, attach the coils, verify the working. Make necessary changes.
Apr 1-3	Add SMA pins, start interfacing with Lock-in
Apr 3-6	Debugging
Apr 7-8	Once interfacing is working, 3-D print the overall box for fluxgate
Apr 9-12	Packaging of all the components

1.7 Link to demo-video

The demo video for the fluxgate can be found at this `drive link`. We perform a sequential testing of all the components, starting from the astable multivibrator all the way to the output of the sense coil.

2 Analog LIA

2.1 Simulations

2.1.1 Analog Phase shifter

The phase shifter is supposed to shift the phase of a reference signal by 90 degrees. The reference signal would ideally be a square wave for the Fluxgate magnetometer demonstration, but for a general lock-in amplifier, a sinusoidal reference suffices. The phase shifter uses an all-pass filter which has a 90-degree phase shift at the exact frequency required, which is chosen to be about 3.2kHz as per the reference generated by the fluxgate magnetometer. The resistance can be changed using a potentiometer located at the position of R_2 in the circuit diagram which follows.
The phase shifter, however, does not generate a 90-degree phase shift for all the frequencies, which creates quite a bit of distortion when a square wave is used as a reference.

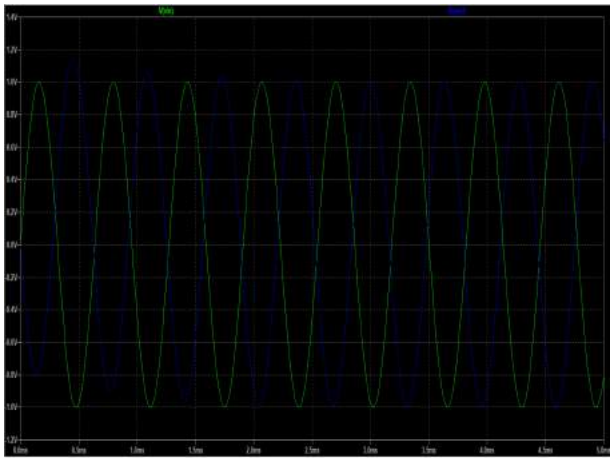
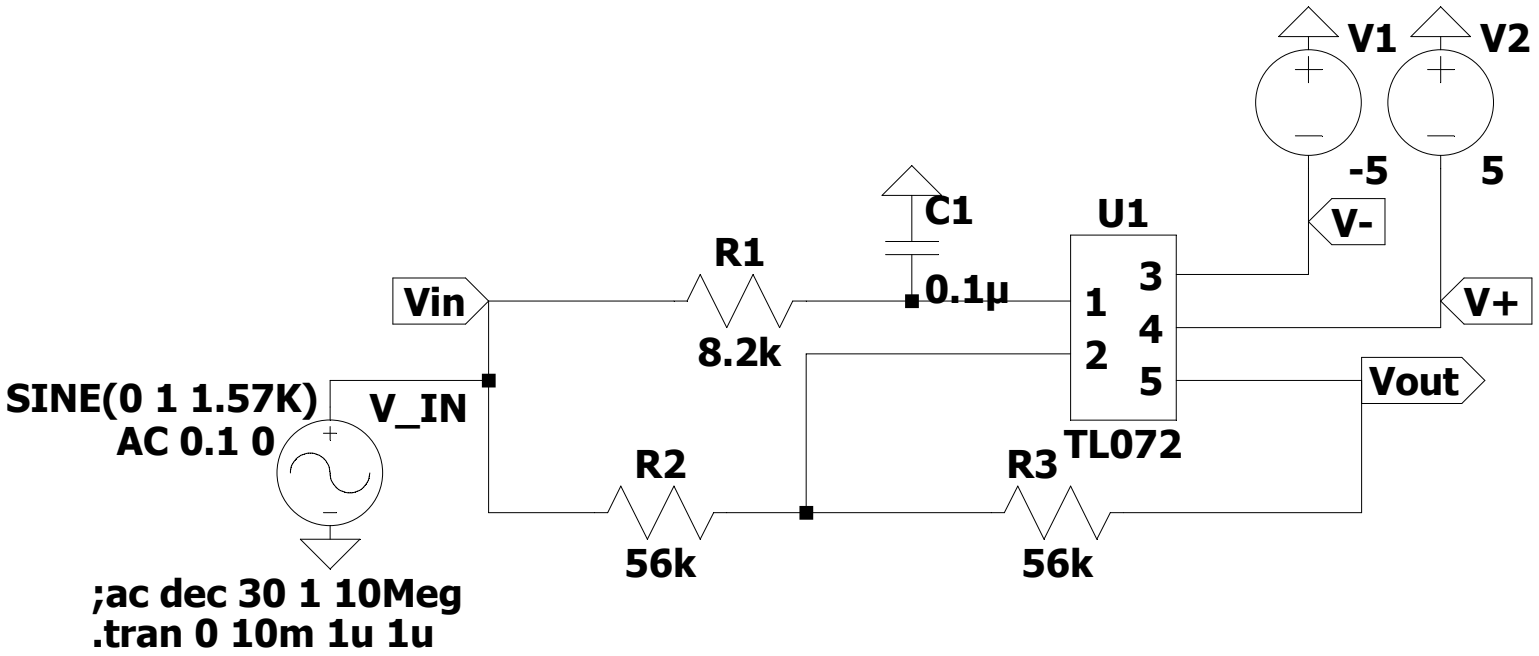


Figure 18: Simulated plot for the phase shifter implemented using an all-pass filter, desired to perform a quadrature phase shift at a predetermined frequency

Initially it was decided to design the phase shifter using the ADG8510 IC which is also an op-amp, but on further comparisons and analyses, it was discovered that TL072 which was readily available in WEL, would also be a good fit. Thus, changes were made in the prior design and for consistent results as well as countering parasitic resistances, a potentiometer was added at the position of R2 in the circuit diagram below. Satisfactorily in-quadrature signals were obtained after testing this setup.

Here is the final circuit schematic from LTspice on which simulations were performed:

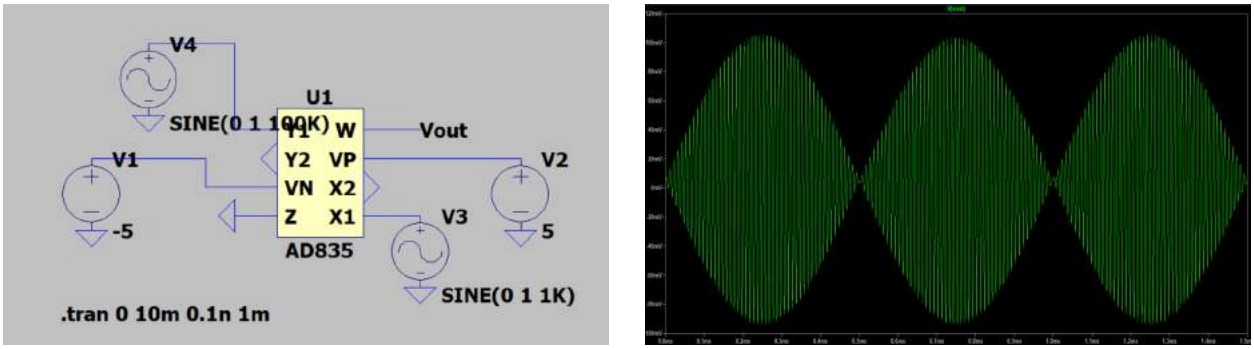

V1
V2
-5
5
V-
V+
C1
0.1µ
U1
R1
8.2k
Vin
1
2
3
4
5
TL072
Vout
SINE(0 1 1.57K)
AC 0.1 0
V_IN
R2
56k
R3
56k
;ac dec 30 1 10Meg
.tran 0 10m 1u 1u

2.1.2 Mixer

We initially planned on using an analog multiplier for multiplying the reference and the noisy signals. We had selected the bipolar analog multiplier AD835 for this purpose. AD835 takes in 2 signals X and Y (as X1, X2, Y1 and Y2), multiplies the 2 signals and adds an offset Z to the product. Basically performing,

$$W = (X1 - X2)(Y1 - Y2) + Z \tag{1}$$

We used a sine wave of frequency 1KHz and another of 100KHz for multiplication. The results of the simulation are shown in the figure below. The multiplication of the 2 sine waves was carried out without any errors. However, analog multipliers are expensive, and we had to look for a different implementation of multipliers. For this purpose, we used a switching circuit along with an instrumentation amplifier.



(a) Circuit for simulating AD835 (b) Results obtained on multiplying 2 sine waves

Figure 19: AD835 Simulations

Further, as the IC AD835 was found to be expensive, we tried to find lesser costly alterrnatives. We found that a mixer could be implemented using an SPDT switch and an instrumentation amplifier as well. HEF4066BP, an SPST IC was available in the lab, as was INA128, an instrumentation amplifier. To implement the aforementioned SPDT switch, we used an inverter(CD74HC04) IC and two SPST switches to model an SPDT switch. Thus the final mixer design has one inverter, two SPST switches and one instrumentation amplifier.

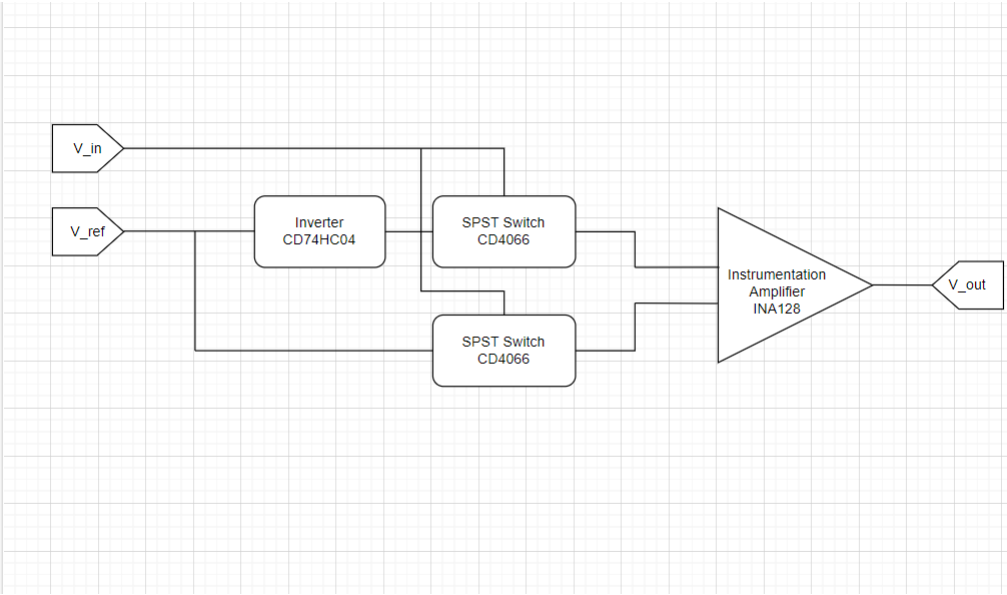
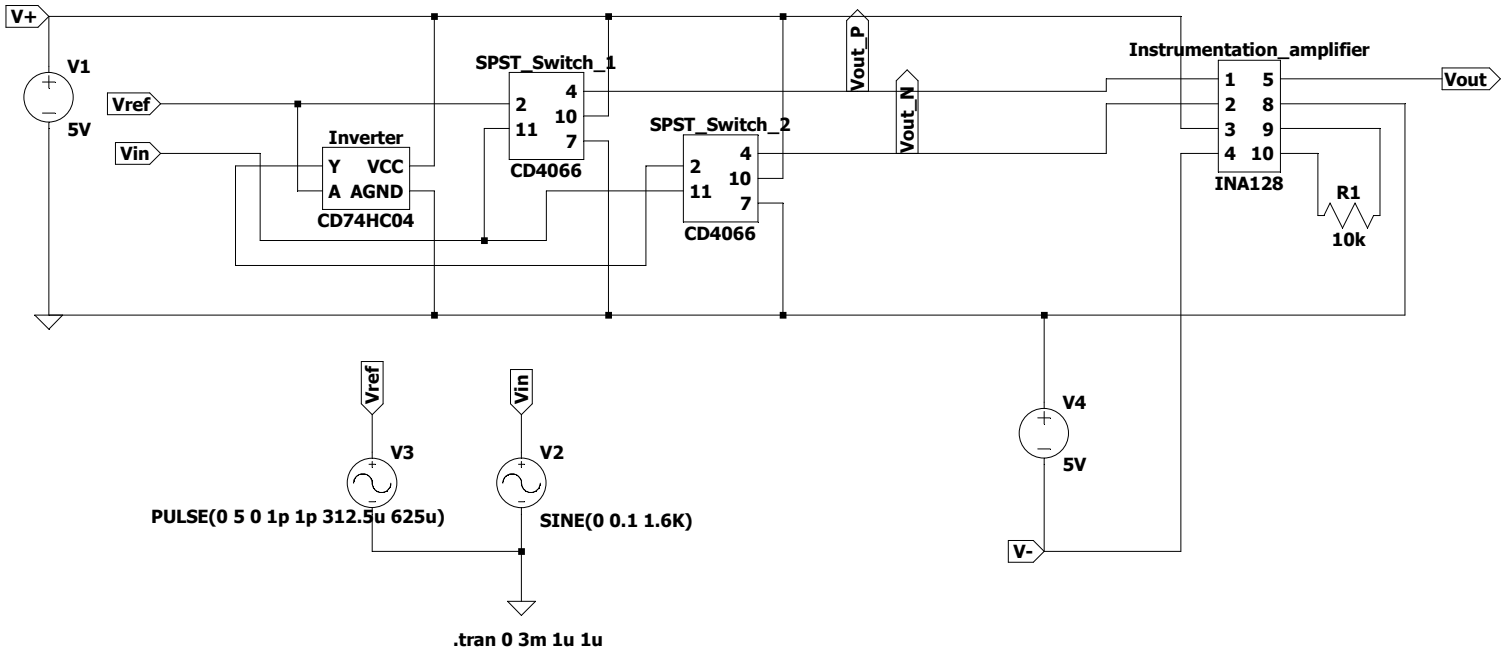
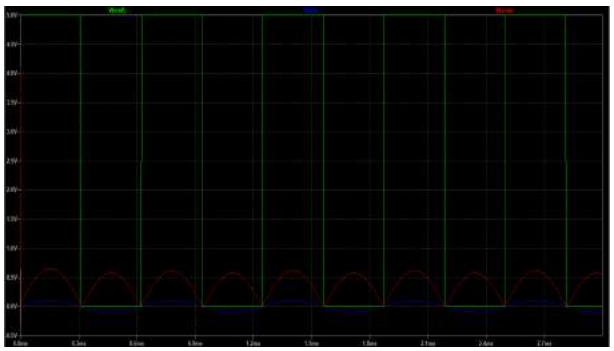
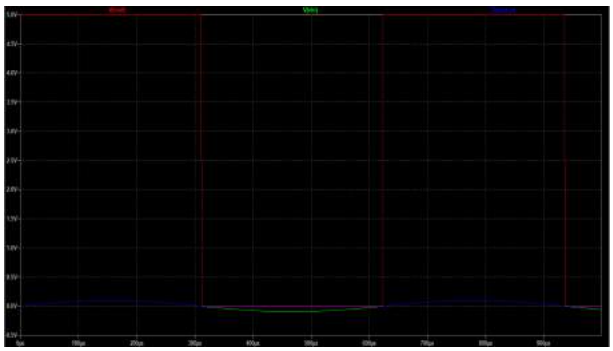


Figure 20: Diagram of mixer implementation: Switching circuit + Instrumentation amplifier

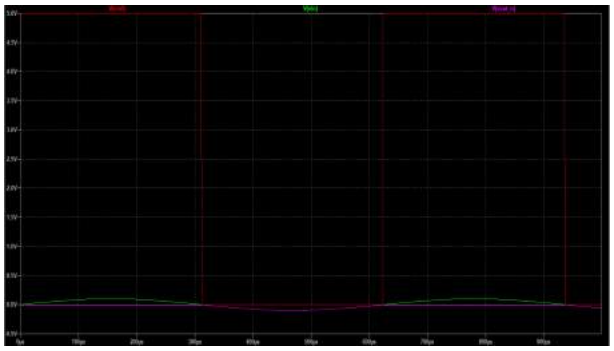

V+
V1
5V
Vref
Vin
Inverter
Y VCC
A AGND
CD74HC04
SPST_Switch_1
2 4
11 10
7
CD4066
SPST_Switch_2
2 4
11 10
7
CD4066
Vout_P
Vout_N
Instrumentation_amplifier
1 5
2 8
3 9
4 10
INA128
Vout
R1
10k
Vref
V3
PULSE(0 5 0 1p 1p 312.5u 625u)
Vin
V2
SINE(0 0.1 1.6K)
V4
5V
V-
.tran 0 3m 1u 1u



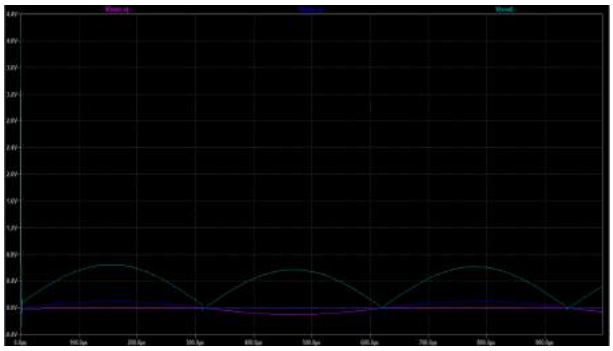
(a) Output versus input and reference signals



(b) Output of SPST with in-phase reference input



(c) Output of SPST with inverted reference input



(d) Inputs and output of instrumentation amplifier

Figure 21: Mixer Simulation results

In the above simulations we can see that the signals Vref and Vin are multiplied and the result is Vout. The signal Vin is taken as a sinusoid here just for reference, but for the flux-gate demonstration it will be a square wave. There will be two such multipliers used finally, one with phase shifted reference and one without any phase shift.

2.1.3 Low pass filter

After multiplication of the two signals, high frequency components have to be filtered out, for which we have used a simple R-C filter along with a TL071 op-amp. Here is the AC analysis output for the same:

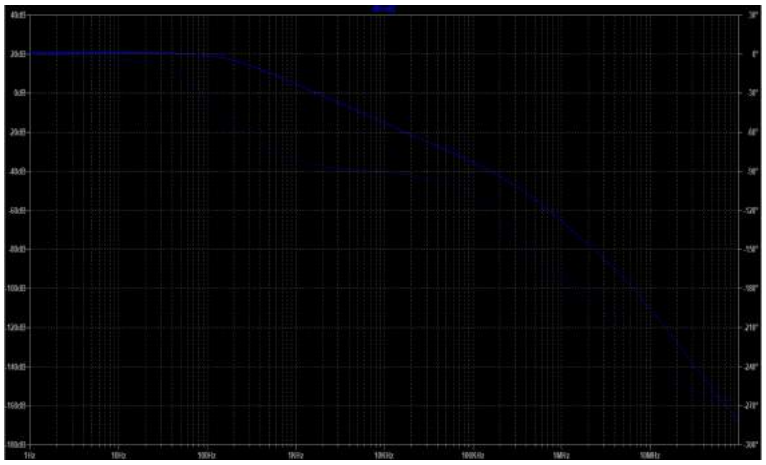
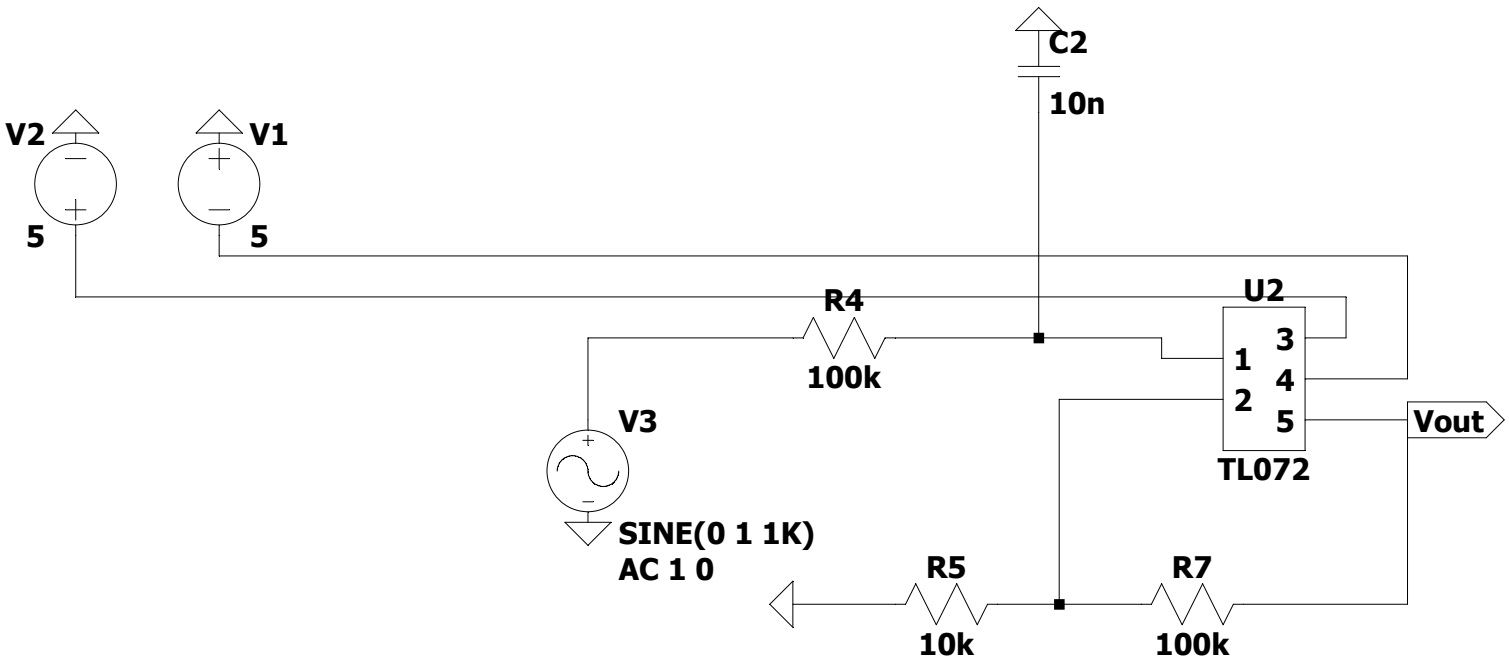


Figure 22: R-C Low-Pass filter

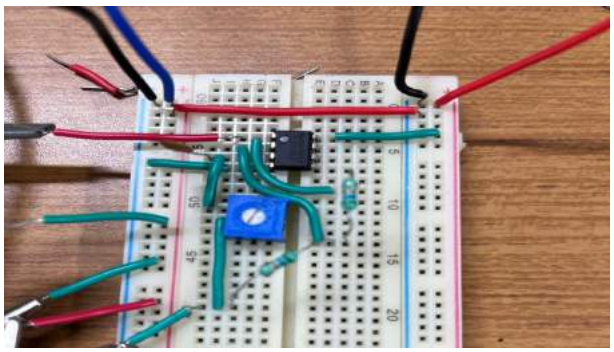


.ac dec 10 1 100Meg

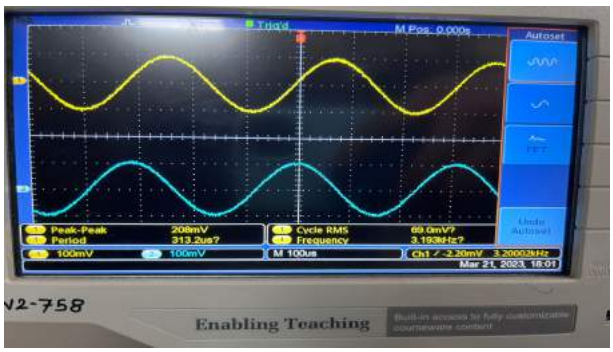
2.2 Analog LIA breadboard testing

2.2.1 Analog Phase Shifter

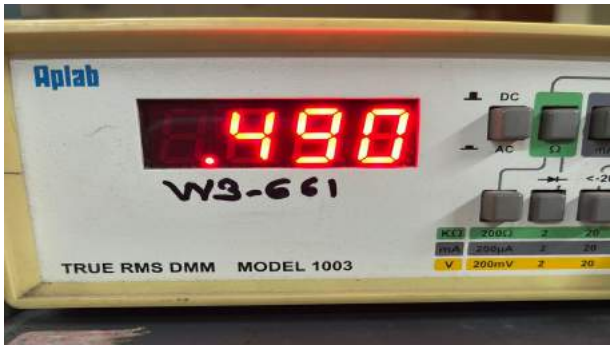
Here are results from the breadboard testing of the phase shifter. As visible in the first image, the setup has a (10k)potentiometer in the position of R2 in the priorly discussed schematic. Tuning the potentiometer led to a good enough phase shift of almost 90 degrees as visible in the second image. The value of resistance R2 at which this phase shift occured is shown in image 3. Finally, the distorted output when a square wave is given as input is shown in image 4.



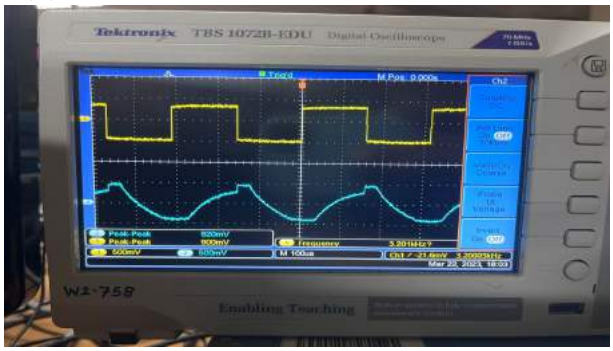
(a) Setup for the phase shifter



(b) 90 degree phase shifted input and output



(c) value of resistance of potentiometer segment



(d) Output for a square wave input

Figure 23: Phase shifter breadboard testing results

Note that the yellow signal in both images is the input and the blue signal is the output.

2.2.2 Mixer

Below is the breadboard setup for one mixer circuit. The three ICs used are an inverter, an SPST switch IC and an instrumentation amplifier. Vref is a square wave sent to the SPST switches as a control signal, Vin is the analog signal input to the SPSTs and Vout is the output of the instrumentation amplifier.

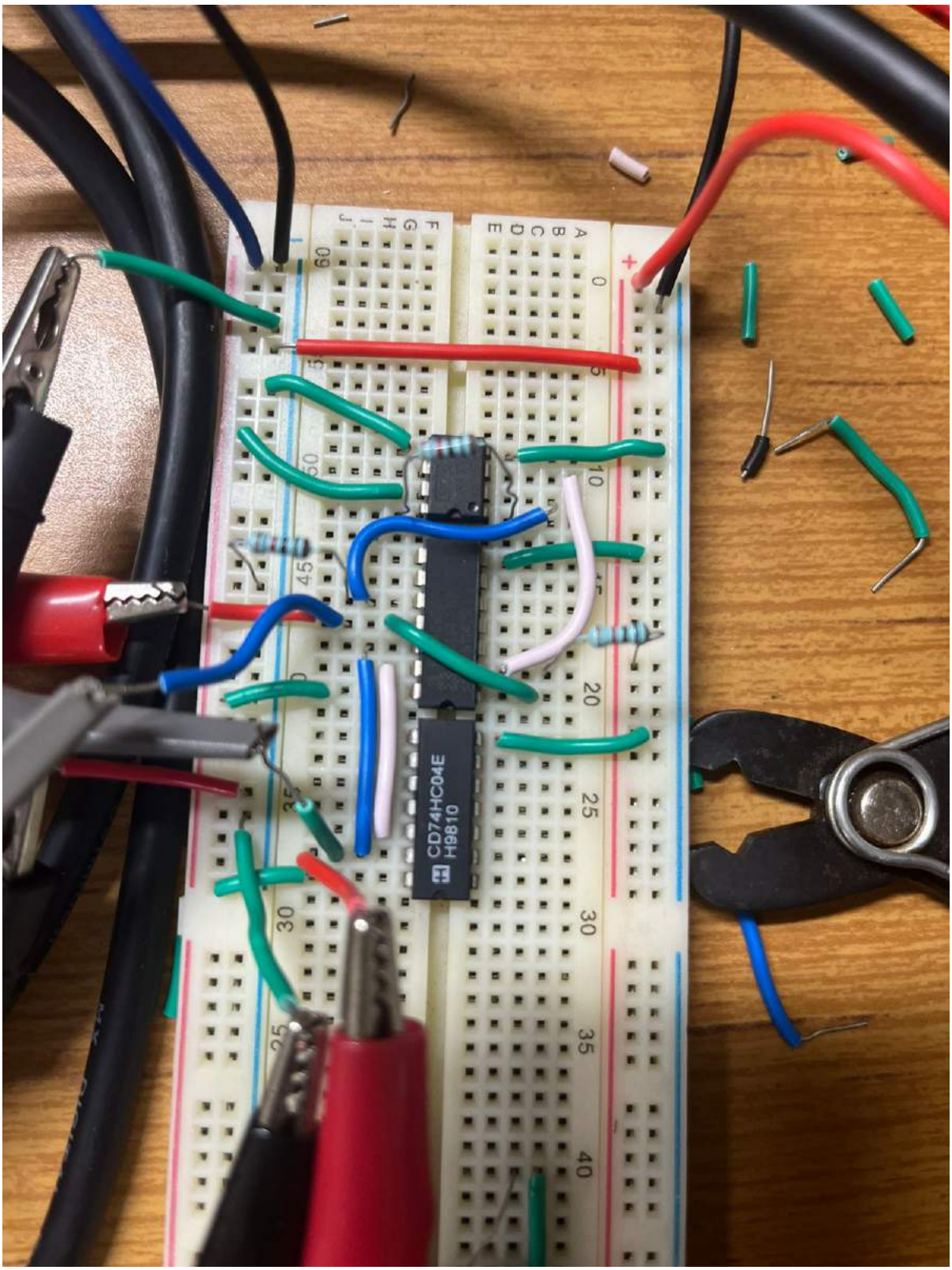
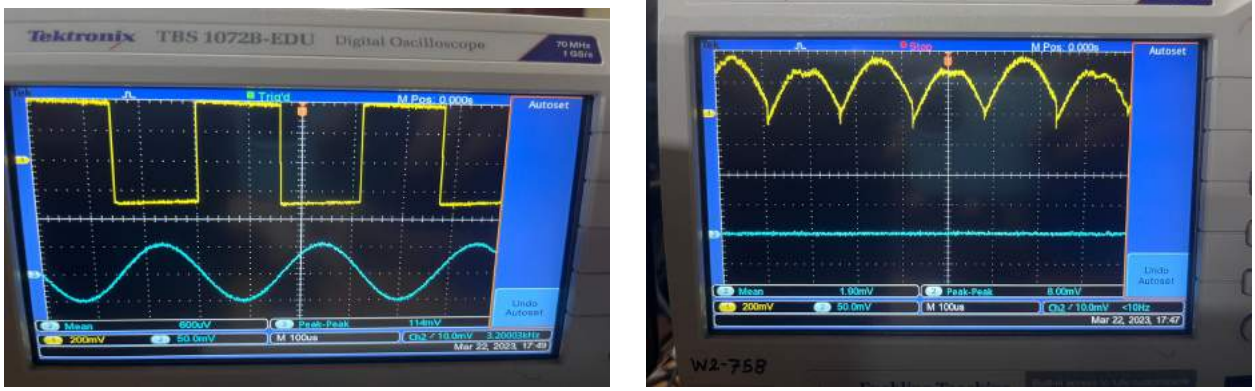


Figure 24: Breadboard testing setup of the analog mixer circuit

Here are the results obtained from this circuit:



(a) Inputs of the mixer (Yellow: Vref, Blue: Vin) (b) Output of the mixer

Figure 25: Analog mixer breadboard testing

As there would be two mixers, one of which would have a perfect square wave as Vref and the other would have the signal obtained as output of the phase shifter when fed with Vref, there is significant distortion. To observe this, the circuit in the first image in the following figure was set up on a breadboard. The resulting output, along with the distorted reference voltage input to the mixer, are also shown alongside.

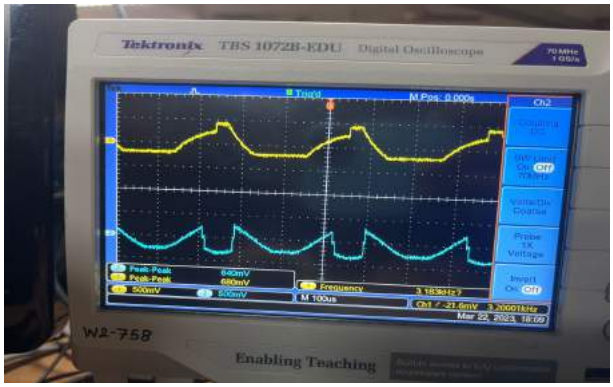


Figure 26: Phase shifted reference and Output of the mixer

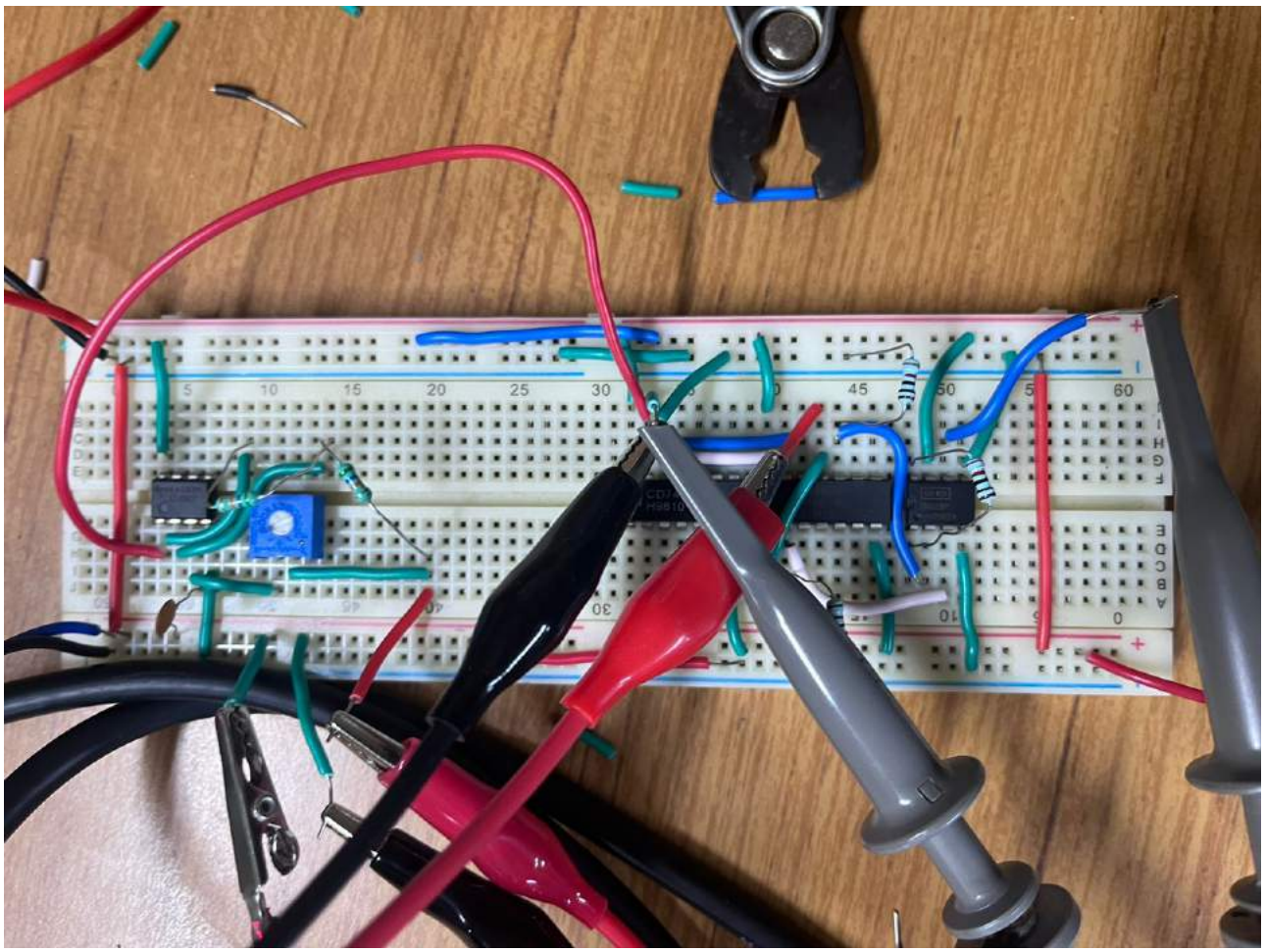
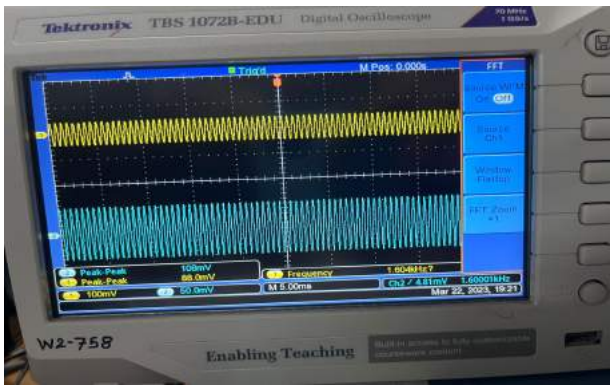


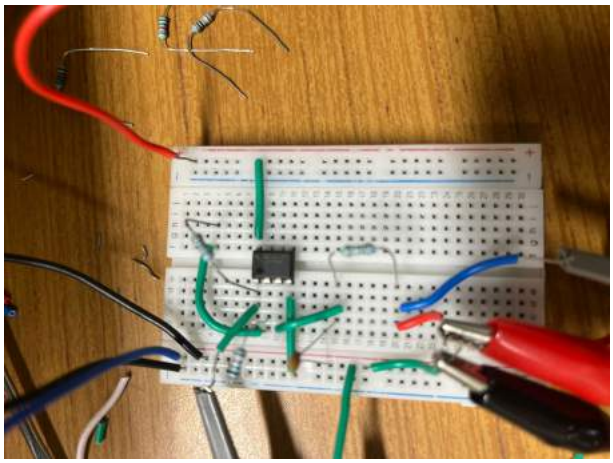
Figure 27: Breadboard setup for phase-shifter and mixer together

2.2.3 Low pass filter

Here is the low pass filter's breadboard setup and output at 3.2kHz input frequency. The values of resistances and capacitors were chosen so that there is significant attenuation at 3.2kHz, as seen in the output itself



(a) Inputs of the filter (Yellow: Vout, Blue: Vin)



(b) Filter breadboard setup

Figure 28: Low pass filter breadboard testing setup

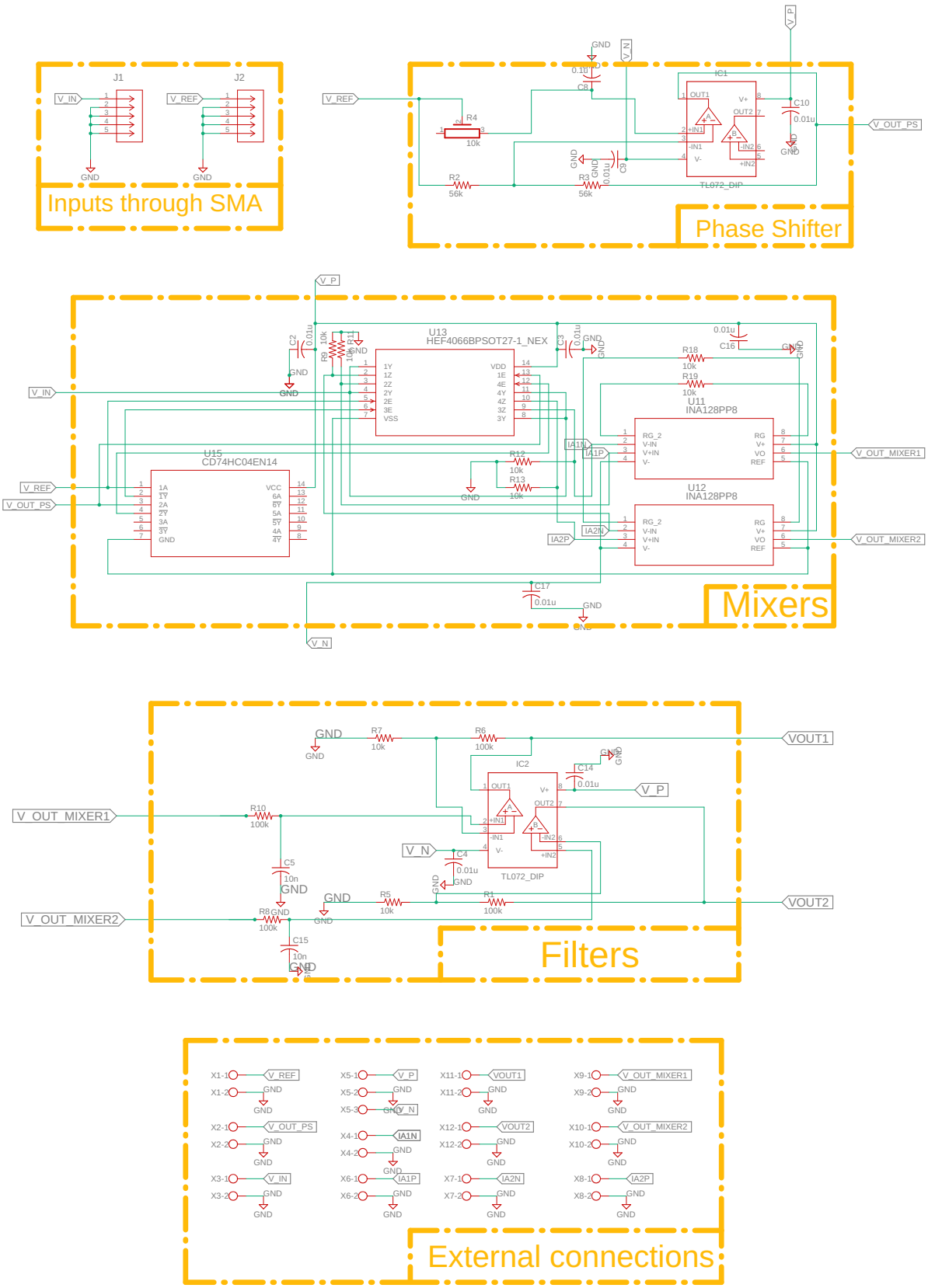
2.3 Microcontroller code

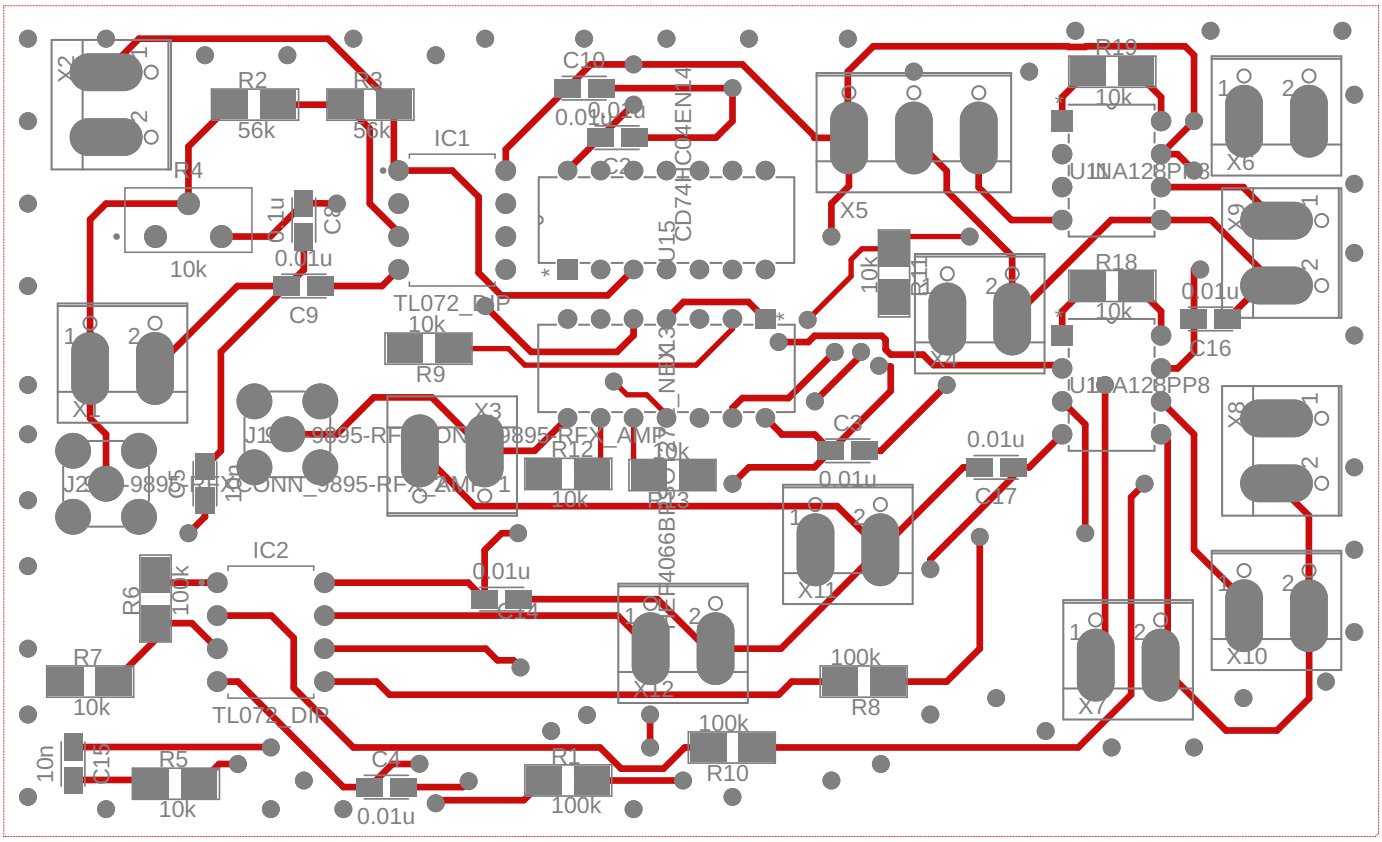
Here is a link to the drive folder which contains the code for ATxMega128, to which the output of the filter will be sent. The ADCs will take in two signals and then store samples, square them element by element, add the two lists to each other, and then take the square root of each element. This will be sent through UART for observing. The UART interface uses the USB CDC, the code for which has been taken from an example repository in microchip studio. *It is to be noted that this code is not tested on the final output.

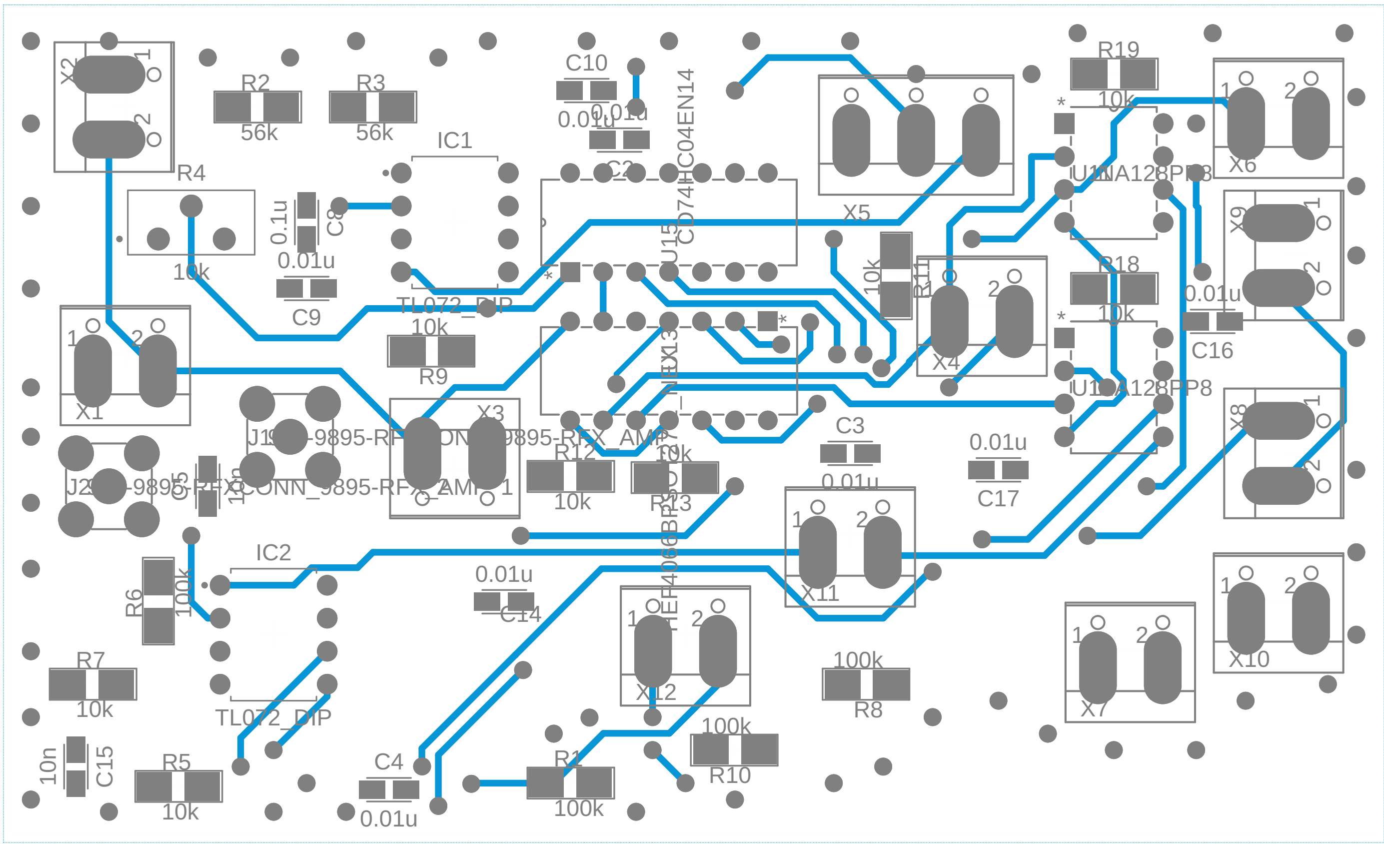
2.4 Circuit Schematic and Layout

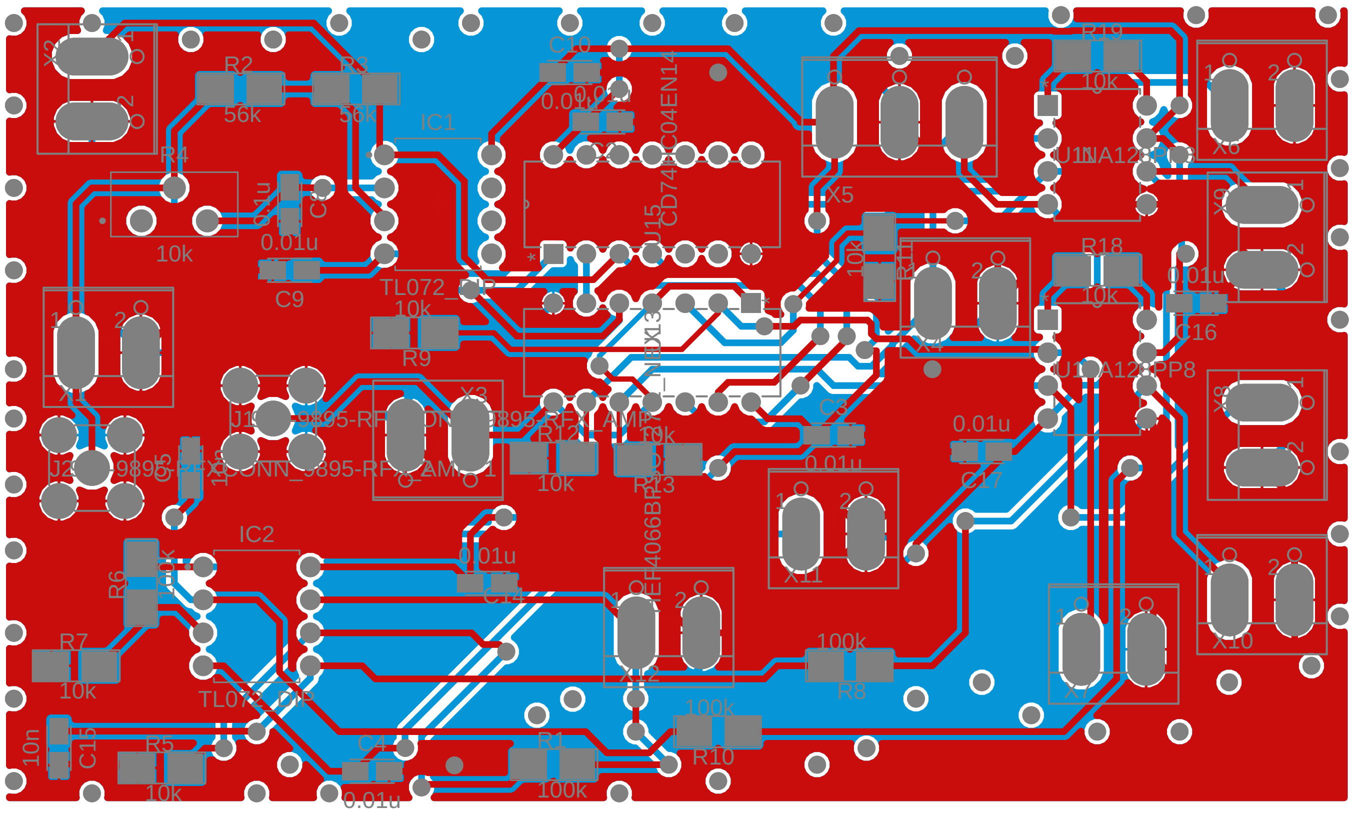
On the following pages, in order, are the circuit schematic, PCB layout and then the top and bottom layers of the PCB. The PCB was checked for DRC errors and none were detected.

Analog Lock-in Amplifier




R2
56k
R3
56k
IC1
C10
0.01u
C2
U15
CD74HC04EN14
R13
10k
U1
INA128PP8
X6
X5
R4
10k
C8
1u
C9
0.01u
TL072_DIP
R9
10k
X1
X4
R18
10k
C16
0.01u
U1
INA128PP8
X8
X3
R12
10k
R23
10k
C3
0.01u
C17
0.01u
X11
IC2
C14
0.01u
R6
10k
R7
10k
TL072_DIP
X12
R8
100k
X7
X10
C13
10n
R5
10k
C4
0.01u
R1
100k
R10
100k


R2
56k
R3
56k
C10
0.01u
IC1
R4
10k
C9
0.01u
TL072_DIP
R9
10k
X5
R19
10k
R18
10k
C16
0.01u
X1
X2
X3
X4
X6
X7
X8
X9
X10
X11
X12
IC2
TL072_DIP
R6
100k
R7
10k
R5
10k
C15
10n
C4
0.01u
R1
100k
R10
100k
R8
100k
R12
10k
R13
10k
C3
0.01u
C17
0.01u
C14
0.01u
C8
0.1u



26-03-2023 18:54 f=2.43 C:\Users\NEERAJ~1\AppData\Local\Temp\Neutron\ElectronFileOutput\15408\brd-46c7514b-f8c4-41e3-9d77-385e8e07d891\Schematic_ALIA v17 v3.brd

2.5 Test results and observations

Here is the overall block diagram for the ALIA. Vin is a general signal but Vref is supposed to be a square wave as it is the control signal for the SPST switches. However in the case of fluxgate magnetometer demonstration, both Vin and Vref are square waves. All the signals between the blocks can be measured as testpoints are designed in the PCB.

- The major possible point of failure is the phase shifter, as it will create distortions in the square wave when sent through it, as seen before. This will propagate throughout and can cause significant effects.
- The inverter has no issues and works perfectly.
- The SPST switch used to have inaccurate outputs but then after adding pull-down resistors at the output, they work correctly.
- The instrumentation amplifier also works perfectly and does not provide any cause for concern.
- Finally, the microcontroller should also not have any issues but since the code is not yet tested on an actual input, it cannot be claimed with absoulte certainty.

Here are the percentage confidences in testing for each individual block:

- Phase shifter: 50%. The phase shifter works perfectly fine if the input is a sine wave. In case of a square wave input however, it shows distortion.
- Inverter: 100 %. The inverter has shown no cause of concern yet.
- SPST Switches: 90 %. The switches as well have not shown any signs of possible errors after having pull-down resistors.
- Instrumentation amplifier: 100 %. The Instrumentation amplifier has shown no cause of concern yet.
- Microcontroller: 70 %. The code has no errors, howver it is yet to be tested on actual inputs, and there might be possible points of error detected when debugging.

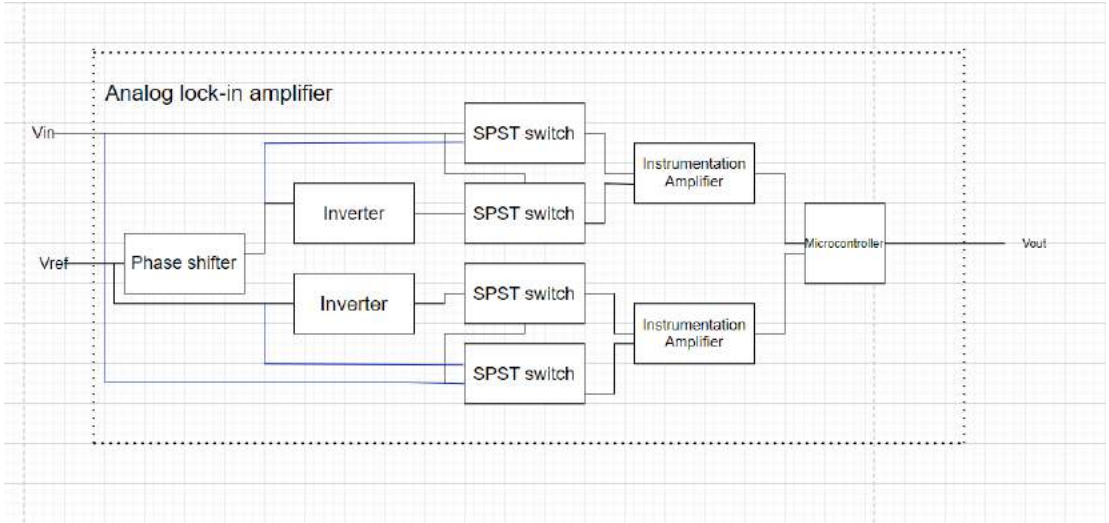


Figure 29: Block diagram for the analog lock-in amplifier

2.6 Next steps

The further steps in the analog lock-in amplifier part of the project are as follows:

- Send the PCB for manufacturing
- Verify the microcontroller code for Analog lockin amplifier and test it with an actual analog input
- Test the fabricated PCB with different inputs
- Test the final ALIA with the flux-gate magnetometer's outputs
- Error-neutralization and debuggin
- Final boxing and packaging

Here is an approximate day-by-day breakdown:

Day	Task
Mar 27-28	Send the PCB for printing and test/debug the microcontroller code
Mar 29-31	After receiving the PCB, test the PCB on inputs generated from the AFG, validate all subcircuits' functioning.
Apr 1-3	Once the Fluxgate magnetometer PCB is tested, test the analog lockin amplifier using the signals provided by the fluxgate magnetometer.
Apr 3-6	Debugging
Apr 7-12	Final 3-D printing, boxing and packaging. Making the presentation.

2.7 Demonstrations

Here is the link for demonstration of the phase shifter and the mixer with appropriate outputs and inputs provided:
Link: `drive link`

3 Digital LIA

The Red Pitaya OS consists of multiple options such as signal generation at the RF output and oscilloscopes to measure the signal at the RF inputs. It also includes various DSP options to to visualize signals and their fourier transforms. The Red Pitaya OS also consists of an SCPI server (which can be enabled from the interface), which we have used for programming the Lock-In Amplifier onto the board. We were originally planning on using the PyRPL library. However, the IQ module of the PyRPL library uses an internal reference signal while we need an external one. Hence, we wrote a new code using the `redpitaya_scpi` library.

Figure 30: Red Pitaya OS

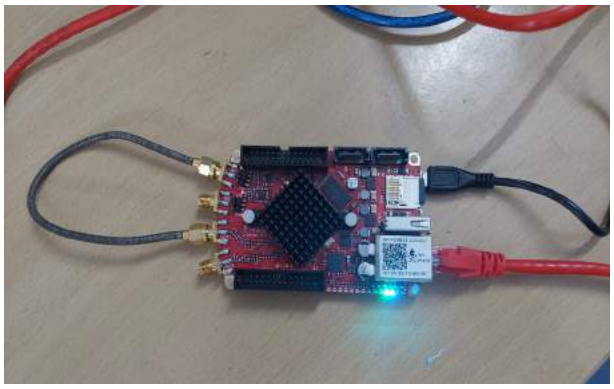
The Red Pitaya board can be controlled remotely over an Ethernet connection using Python via the Red Pitaya SCPI list of commands. SCPI uses a set of SCPI commands that are recognised by the instruments to enable specific actions to be taken (e.g., acquiring data from fast analog inputs, generating signals, and controlling other peripheries of the Red Pitaya platform). The Python code provides powerful data analysis tools, and SCPI commands simple access to raw data acquired on the Red Pitaya board.

3.1 Codes

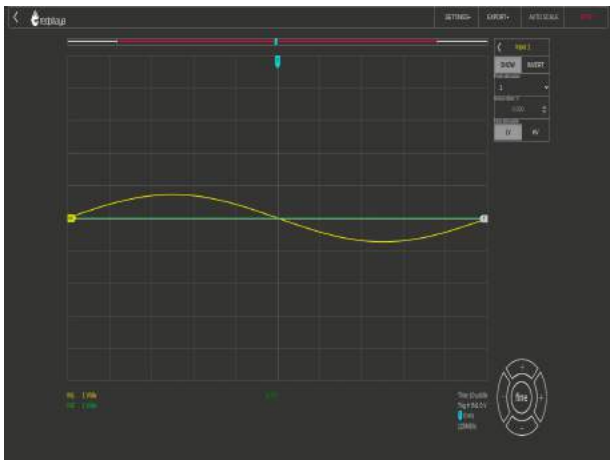
PyRPL Library

```
from pyrpl import RedPitaya
r = RedPitaya(hostname="10.42.0.245")
r.asg0.setup(waveform='sin',
             amplitude=0.7,
             frequency=10e3,
             output_direct='out1')
```

The above code snippet connects to the Red Pitaya board with IP address "10.42.0.245". Then it generates a sine wave of amplitude 0.7 and frequency 10KHz. For testing this code snippet, the output at the DAC channel out1 was looped back and connected to the ADC channel in1. The result is shown in the figure.



(a) Loop back from out1 to in1 channels on the Red Pitaya board



(b) The input waveform at in1. As can be seen, the amplitude of the waveform is 0.7V and the frequency is 10KHz.

Figure 31: Signal Generation on Red Pitaya

```
iq0=r.iq0
iq0.setup(frequency=3e3, gain=0.0, phase=0, amplitude=0.5,
          output_direct='out1', output_signal='quadrature',
          quadrature_factor=10)
```

This uses the IQ module 0 of the red pitaya and sets the demodulation frequency to 3KHz. For the other IQ module, the phase can be changed to 90. To eliminate any phase offsets between the 2 IQ modules, the command `synchronize_iqs` is used.

Red Pitaya SCPI

The SCPI library consists of multiple functions and commands to read the digital, analog and RF inputs while also being able to set outputs at the respective ports. The lock-in input signal and the reference signal will be given to the 2 RF input ports. The output from the lock-in amplifier, which is the amplitude of the signal, will be transmitted to the PC using the ethernet connection. The setup is shown in the diagram:

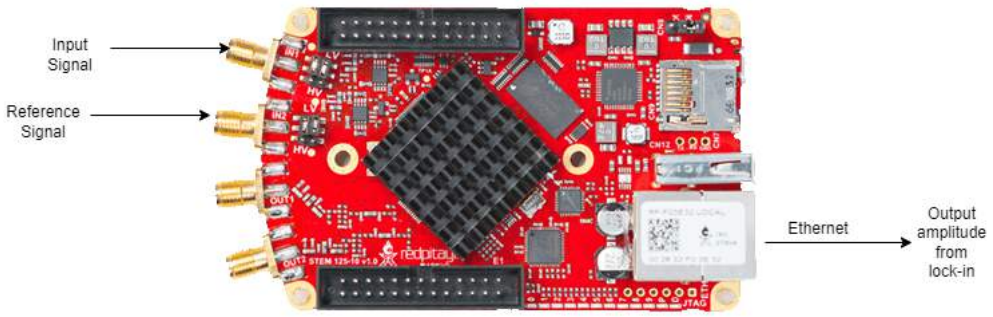


Figure 32: Setup for Digital Lock-In Amplifier

While reading the RF inputs, the functions in the SCPI library return the signal at multiple sampling frequencies, which can be programmed. The table here gives details about the sampling frequencies for each decimation value. For testing purposes, we have used a decimation value of 4, which gives us a sampling rate of 31.25 MS/s. Since the frequency of our signal is approximately 3.2 KHz, the sampling frequency is more than enough for the experiment right

now. Since the SCPI library in one command returns approximately 16,000 samples from the RF inputs, we have a limited time (approximately 0.5ms) for which values are added into the buffer. We plan to experiment with the decimation further, once we have the entire lock-in setup ready with the fluxgate sensor. This will help us increase the amount of time for which we can capture the data.

```
rp_s.tx_txt('ACQ:SOUR1:DATA?')
buff_string = rp_s.rx_txt()
buff_string = buff_string.strip('{}\n\r').replace("  ", "").split(',')
buff = list(map(float, buff_string))
```

The above code snippet calls certain functions from the `redpitaya_scpi` library which help to read the data from the RF input channel 1. Since the data received is in the form of a command string (SCPI communicates using strings), we separate the entire string into different values, and assign it to a list. This list is then converted to an array of floating point numbers using numpy functions. The array of the reference signal and input signal values hence obtained is then used for further processing.
The hilbert function in the scipy library is used for shifting the reference signal by 90 degrees. This signal is then used for the generation of the quadrature-phase signal. Element-wise multiplication is carried out in python using the 2 reference signals (original and 90-degree phase shifted) with the input signal to generate the in-phase and quadrature-phase components. These signals are then passed through a Butterworth lowpass filter with a cutoff frequency of 5kHz. Since lowpass filters can be implemented in multiple ways on Python, we will be tuning this once we have the final setup. The codes can be executed using the IP address of the Red Pitaya board by running the following command: python3 code.py 10.42.0.245 where 10.42.0.245 is the IP address of the Red Pitaya board.

3.2 Testing

The following test setup was used to test the various functions and processing of the lock-in amplifier code. The signal was generated from the oscilloscope present in the Red Pitaya OS through the RF outputs 1 and 2. These signals were looped back to RF inputs 1 and 2 respectively. These inputs were read using the code explained above and processed further.

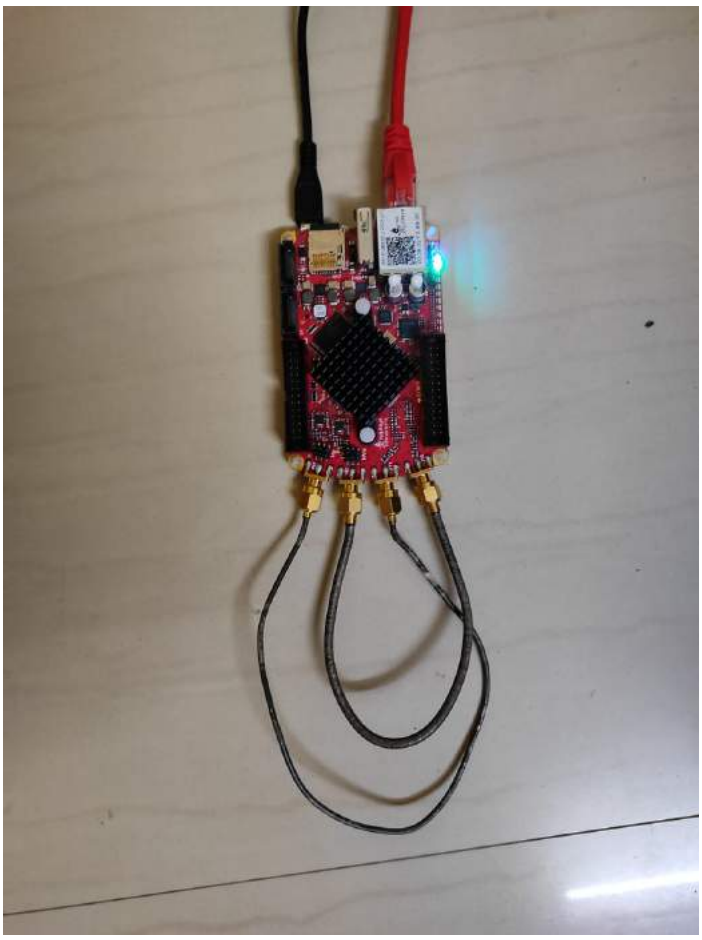


Figure 33: Test Setup for the Red Pitaya board

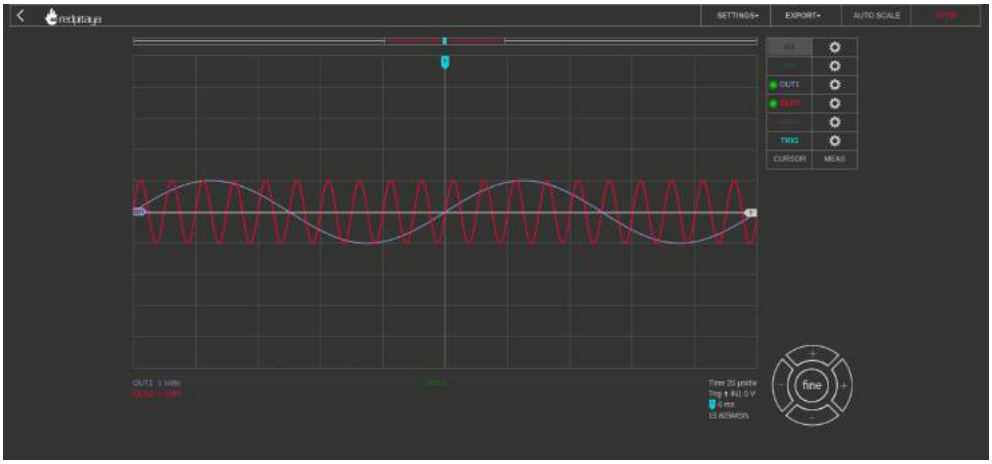


Figure 34: The 2 input signals to the red pitaya board, displayed on the signal generation oscilloscope

To start with, 2 sine waves were looped back as inputs from the RF channels, one with a frequency of 10KHz and one with a frequency of 50KHz. The following results were obtained when these signals were acquired using the above code and displayed using matplotlib:

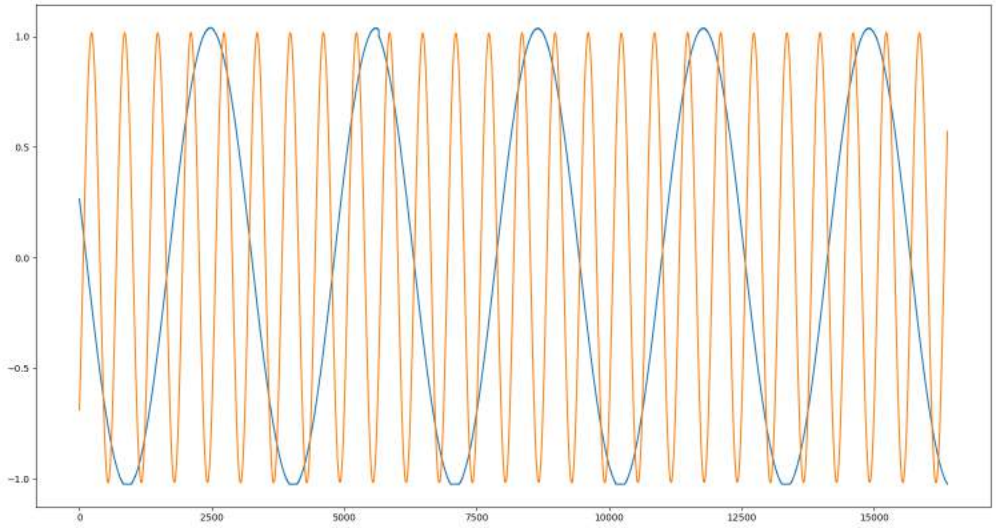


Figure 35: The 2 signals acquired using the codes and displayed. The x-axis shows the sample number and the y-axis shows the amplitude of the signal.

As can be seen, the 2 signals were read almost entirely accurately. However, in very few cases, we also found that there were slight distortions in the signal received. This may be due to multiple reasons:

- Noise added while signal transmission through coaxial cable
- Some issues in signal generation (Possibly due to some reset that occurs after some amount of time): This will be fixed in the final setup as the signal will be coming from an external source
- Some error in the function that acquires the signal (Could be something related to the RF input fluctuations): These fluctuations cannot be entirely dealt with. However, experimenting with the sampling rate and the decimation values can help in reducing this

The hilbert transform was used to generate the 90-degree phase shifted version of the reference signal. The output obtained was found to be accurate for most of the duration of the signal. The hilbert output went very high at the beginning and at the end. This could be due to the working of the scipy function and generation of the fourier transform from where it is calculated. However, this does not bother us because in the final setup, we will be able to find the magnetic field with the remaining data.

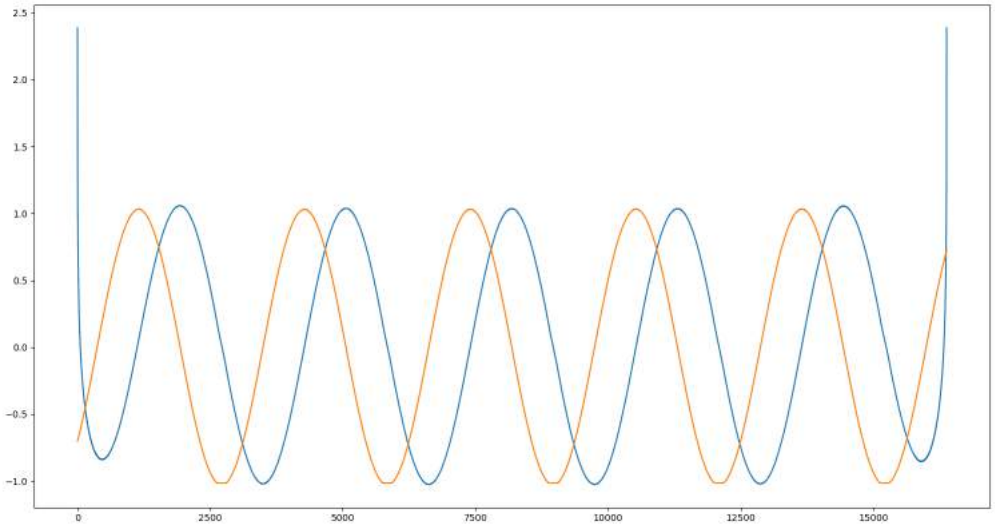


Figure 36: Reference signal and 90-degree phase-shifted reference signal

The following is the output of the multiplication of 2 signals: 50KHz and 10 KHz. As can be seen, the 50KHz wave envelopes the output.

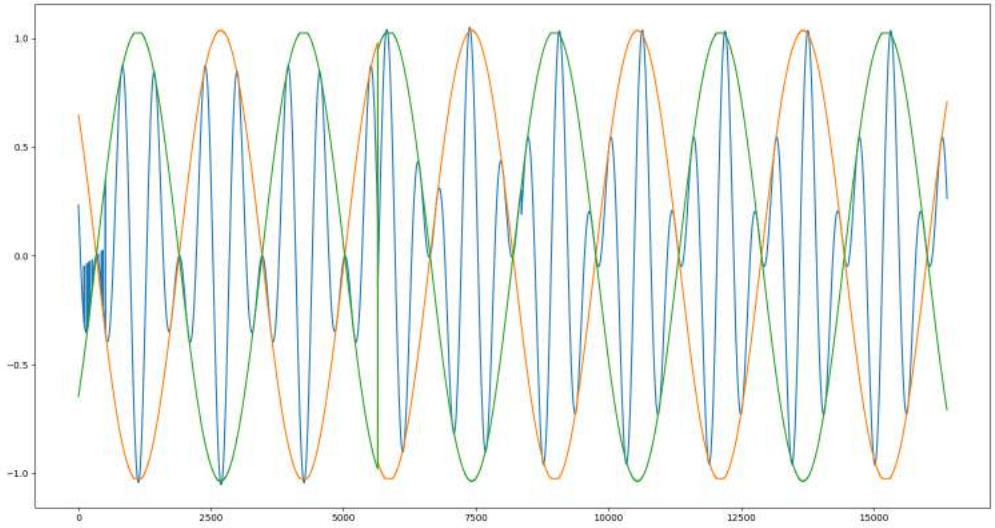


Figure 37: Multiplication output

We also explored the use of a ping-pong buffer to reduce any loss in data. This involves the use of 2 buffers, one of which is actively acquiring data from the RF input when the data from the other buffer is being processed. However, this may not be necessary for our setup since the processing time is very less compared to the signal acquisition time.

3.3 Next Steps

The remaining tasks in the digital lock-in amplifier are as follows:

- Tune the decimation value to increase signal acquisition time while also avoiding loss of data
- Test the entire setup with the inputs from the fluxgate sensor instead of arbitrarily generated signals

- If necessary, implement a ping-pong buffer
- Debug the code and tune any other values if necessary
- We will also compare the PyRPL implementation with the SCPI implementation to find which gives better results
- Final packaging with external interfaces for power and signals

A rough day-wise breakdown is:

Day	Task
Mar 27-28	Tune the decimation value and try to reduce noise
Mar 28-31	Look into the implementation of ping-pong buffer and have a code ready in case necessary while testing
Apr 1-3	Interface with fluxgate and tune any parameter values
Apr 3-6	Debug the code if any errors are found and compare the outputs of the 2 lock-in implementations
Apr 7-9	3D print a box and make external connections for interfacing with external signals

3.4 Link to demo video

The demo video for the digital lock-in implementation can be found at this `drive link`.