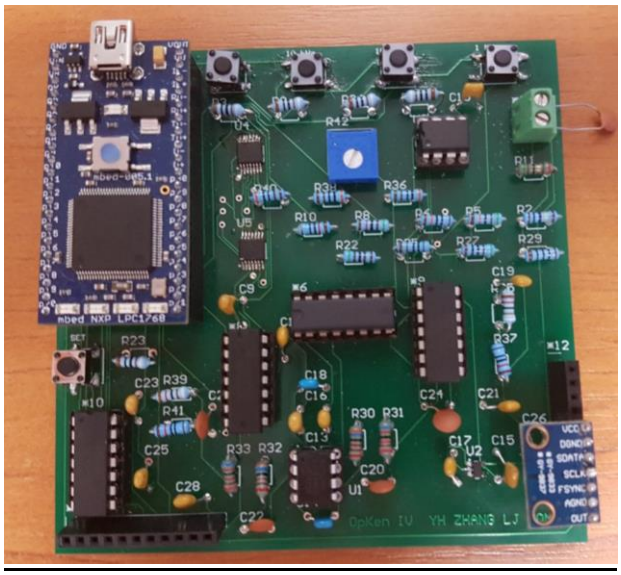

Figure 1: Photograph of Finall PCB

Final Schematic Diagram:

This section aims to highlight the differences between two versions of PCB schematics (refer to documents sent via email).

We have made several significant changes to our schematic since the 2nd PCB with the aim to replace unnecessary analogue components with digital circuits and microcontroller actions. This helps to save PCB area considerably.

Firstly, we removed the switches used to change between different resistors in the auto-balancing bridge and used 2 multiplexers controlled by the microcontroller (p27-p30). We have 8 resistors to select from, so we used two 4x1 multiplexers (ADG1604). The previous idea was to use mechanical push switches, but they are too bulky and does not allow for a smooth operation.

Secondly, we removed the PLL (CD4046). This is because they were incredibly challenging to configure correctly with minimal distortion. In order to perform frequency doubling, the PLL would need to have a set of passive components connected that determined the centre oscillation frequency and frequency response of the loop filter. Since we needed to operate at 4 spot frequencies, this meant that the PLL also needed to double 4 different frequencies, and hence would potentially require 4 different sets of passive components and associated switches. This takes up too much PCB area. To combat this issue, we generated the local oscillatory waveform at twice the intended frequency directly from PwmOut pin of microcontroller. This method gave us reliable square waves up to 2MHz by importing a library FastPWM.h into our main code. Nonetheless, this compromised the synchronization with the signal produced by AD9833 sine wave generator slightly.

Algorithm for calibration, self calibration, automation:

Magnitude computation:

The auto-balancing bridge section of our instrument is capable of self- calibration with the aid of 2 multiplexers, each with 3 control terminals (EN, A0, A1) receiving signals from the microcontroller. The reason for having these is because a total of 9 resistors (8 of which are controlled by multiplexers, 1 always in connection) are in the feedback loop of inverting amplifier of the bridge. Microcontroller BusOut commands for switching of branch with its corresponding values are shown as below.

Microcontroller command (p30:p29:p28:p27)	Range Resistor Value (Ω)
14 (1110)	99.99
6 (0110)	509.74
10 (1010)	999.00
2 (0010)	5074.12
13 (1101)	9900.99
9 (1001)	47619.05
5 (0101)	90909.09
1 (0001)	333333.33
0 (0000)	1000000.00

The auto-balancing bridge should be able to switch between branches under microcontroller commands, starting from the lowest impedance of 99.99Ω to ensure that amplifier does not saturate. The microcontroller then constantly checks for the peak to peak voltage at the output of amplifier, by sampling the signal at 180kHz rate with sample size of 1000 and storing into an array. A 10-point moving average filter was also implemented in the microcontroller code in order to reduce the effects of noise in the output signal. This can be seen in the code snippet below.

```
float x;
for(int i = 0; i < SAMPLE_SIZE; i++){
    adc.start();
    while(!adc.done(p17));
    x = adc.read(p17);
    array[i] = x;
}
//moving average filter
for(int k =0; k<(SAMPLE_SIZE-M+1) ; k++){
          int z = 0;
    for(int q=0; q<M-1;q++){
         z+=array[k+q];

    }
     array_y[k]=z/M;
 }
float min = findmin(array);
float max = findmax(array);
float pk2pk = max - min;
float pk2pk_true = 3.3*pk2pk/4096;//convert this value to 0-3.3V
```

Figure 2: Code snippet for moving average filter and peak to peak detection of output signal.

After obtaining this peak to peak value, the microcontroller checks whether this value lies in the allowable range of 0.20V up to about 3V (a gain range of 1/3 to 5). If this condition is not satisfied, the auto-balancing bridge selects the branch above its current branch by multiplexing actions. The process was repeated until a branch was found for optimal magnitude computation. Thus, impedance magnitude of the unknown DUT could then be calculated by using the formula for gain of an inverting amplifier:

$$Z_x = R_r \frac{V_x}{V_r}$$

,where Z_x is the impedance of unknown DUT, R_r the value of resistance in the branch where the selection process stops, V_x the peak-to-peak value of source signal which is ≈660mV when driving a high impedance load and V_r the peak-to-peak value measured at the output. The following flow chart summarises the process of self-calibration for magnitude computation.

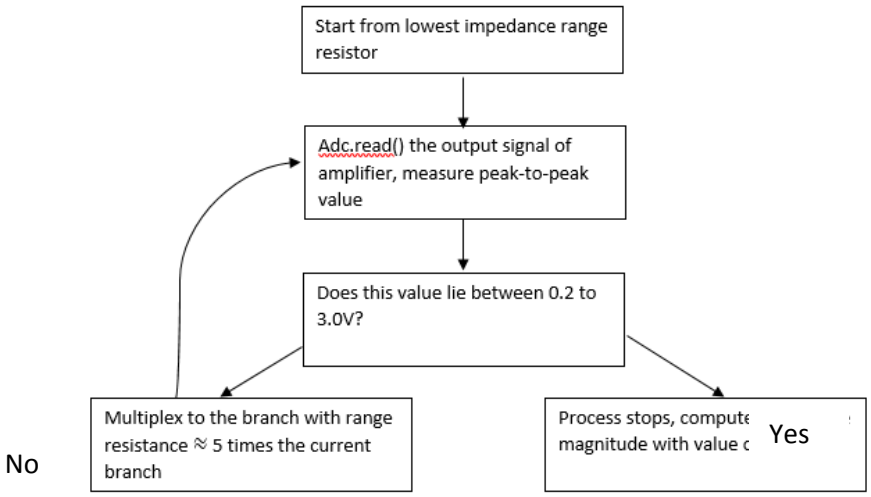

Figure 3: Flow chart for self-calibration of auto-balancing bridge

Phase computation:

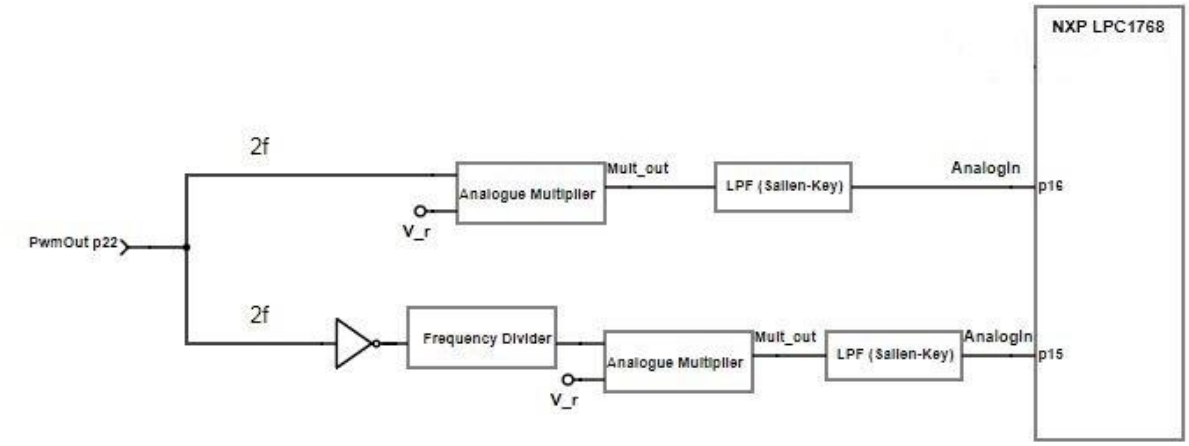

Figure 4: Phase computation, with frequency multiplier(PLL) removed

We did not manage to get this part of circuit work as expected but we have identified some potential solutions. The main problem was that the peak-to-peak signal at output of analogue multiplier would eventually saturate (i.e. > 3.3V) the Sallen-Key filter (3.3V single rail), which made the detection of DC value at the analogue input pins impossible. To potentially combat this problem if given more time, we would average the input signals entering the multiplier input pins in order to annul the DC offset of incoming signals. This can be done since the transfer function of multiplier is in the form of $(x-x_0)(y-y_0)$, where x, x_0, y and y_0 are input pins to our analogue multiplier. Since the signals feeding into AnalogIn pins are now bipolar, further actions would be required in order to bring them in the microcontroller's

readable range of 0-3.3V. Nonetheless, we wrote codes which computed the phase in all four quadrants by using atan() function of the math.h library.

```
float sum1 = 0;
for(int i = 0; i < SAMPLE_SIZE; i++){
    adc.start();

    while(!adc.done(ADC1));
    in_phase= adc.read(ADC1);
    sum1 = sum1 + 3.3*in_phase/4096;
}
in_phase = sum1/SAMPLE_SIZE;

adc.select(ADC2);//in phase component

float sum2 = 0;
for(int i = 0; i < SAMPLE_SIZE; i++){
    adc.start();

    while(!adc.done(ADC2));
    quadrature= adc.read(ADC2);
    sum2 = sum2 + 3.3*quadrature/4096;
}
quadrature = sum2/SAMPLE_SIZE;
if((quadrature-1.65) > 0 && (in_phase-1.65) <= 0){ //2nd quadrant
    phase = PI+atan((quadrature-1.65)/(in_phase-1.65));
}

else if((quadrature-1.65) < 0 && (in_phase-1.65) < 0){  //3rd quadrant
    phase = atan((quadrature-1.65)/(in_phase-1.65))-PI;
}
else{
    phase = atan((quadrature-1.65)/(in_phase-1.65));
}
```

Figure 5: Code snippet to compute phase, with assumption that our reference 'zero' lies in 1.65V

List of Input and Output Signals:

DUT	1k				10k				100k				Branch	Ad9833 pk-to-pk	relative error		
	Magnitude	impedance calculated	true phase	Phase	Magnitude	impedance calculated	true phase	Phase	Magnitude	impedance calculated	true phase	Phase			1k	10k	100k
51	1.61	55.40652174	-180	-180	1.59	56.10345912	-180	-180	1.58	56.4585443	-180	-180	509.74	0.175	0.0864	0.1001	0.107
180	1.28	114.6915	-180	-180	1.22	120.3320656	-180	-180	1.22	120.3320656	-180	-180	509.74	0.288	0.3628	0.3315	0.3315
510	0.8	270.799375	-180	-180	0.77	281.35	-180	-180	0.77	281.35	-180	-180	509.74	0.425	0.469	0.4483	0.4483
1200	2.09	1177.471158	-180		2.09	1177.471158	-180		2	1232.75236	-180	-180	5074.12	0.506	0.0188	0.0188	0.0273
3000	0.98	2926.162245	-180		0.97	2956.854639	-180		0.94	3052.85	-180	-180	5074.12	0.575	0.0246	0.0144	0.0176
5100	0.63	4733.170286	-180		0.63	4733.170286	-180		0.59	5057.520814	-180	-180	5074.12	0.594	0.0719	0.0775	0.0083
9100	0.41	7448.796878	-180		0.38	8040.886105	-180		0.38	8040.886105	-180	-180	5074.12	0.606	0.1815	0.1164	0.1164
30000	1.05	28833.05357	-180		1	30277.25625	-180		0.56	54106.60045	-50.4	1.4	48525.21	0.625	0.0389	0.0092	0.8036
51000	0.64	46932.97655	-180		0.63	47626.94443	-7.2	2	0.39	76967.21792	-39.6	1.1	48525.21	0.619	0.0797	0.0661	0.5092
75000	0.47	63238.26326	-180		0.44	67553.4403	-17.28	4.8	0.27	110119.199	-61.2	1.7	48525.21	0.613	0.1568	0.0993	0.4683
91000	0.41	73920.35671	-180		0.38	79760.20066	-12.24	3.4	0.23	131810.9837	-49.68	1.38	48525.21	0.625	0.1877	0.1235	0.4485
110000	1.89	111638.2659	-180		1.08	195405.2153	-54	15	0.22	959461.3295	-56.16	1.56	337748.34	0.625	0.0149	0.7764	7.7224
200000	1.06	195269.5023	-180		0.61	339358.3974	-54	15	0.16	1293947.328	-64.8	1.8	337748.34	0.613	0.0237	0.6968	5.4697
300000	0.75	281405.95	-180		0.42	502550.6964	-51.12	14.2	0.13	1623739.096	-68.2	2.45	337748.34	0.625	0.062	0.6752	4.4125

Specifications achieved and graphs of measurements:

Our instrument works well for resistive DUT of magnitude in the range of 1kΩ up to approximately 7kΩ at spot frequencies of 1kHz, 10kHz and 100kHz as deduced from Fig 6 and Fig 7. This was when range resistance = 5074.12Ω. We also found the higher the closed loop gain of the auto-balancing bridge, the higher the accuracy of impedance of DUT measured, with the relative error going as low as 2%. For phase inspection, we probed the output signal and compared this with the waveform at the signal source. We actually found out that for this range of resistance, the phase at the output of amplifier with reference to signal at source was actually -180 degrees, which was very desirable since our auto-balancing bridge was configured to be an inverting amplifier. The following graphs depict the measurements plotted against true resistive values of DUT.

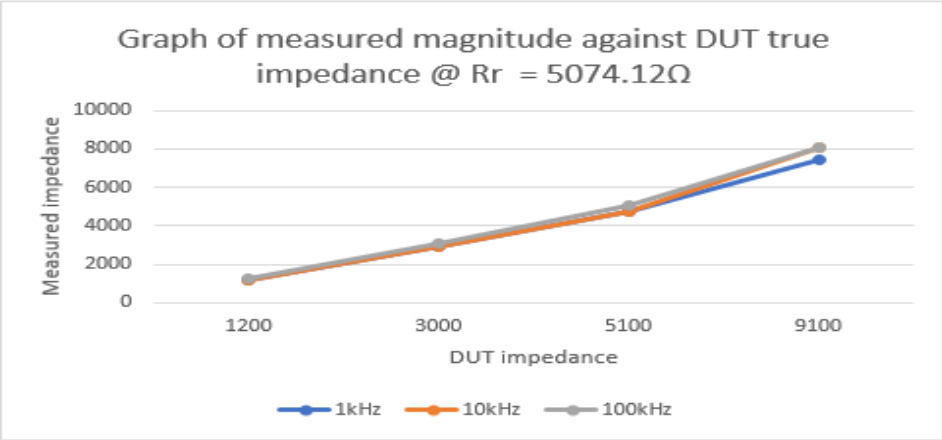

Figure 6: Measured impedance against DUT actual impedance. A linear relationship observed up to 7000 ohms

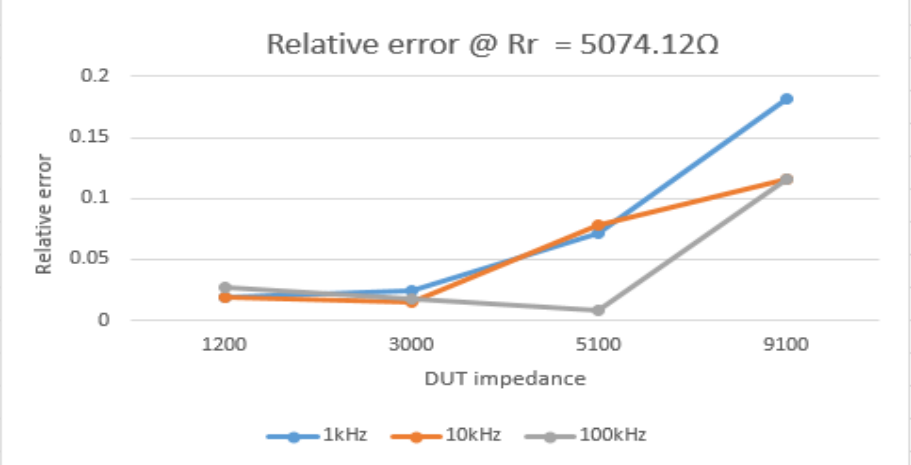

Figure 7: Relative error with reference to actual DUT resistive values

We also tested with low value resistive loads, with range resistor tuned to 509.74Ω. This posed a problem since the AD9833 generator had a relatively high output impedance and hence would have an attenuated version of sine wave loading the DUT. A potential solution, though not implemented on time in our design would be using a buffer to drive low-impedance DUT. Nonetheless, a graph was plotted to inspect its relative error at different spot frequencies.

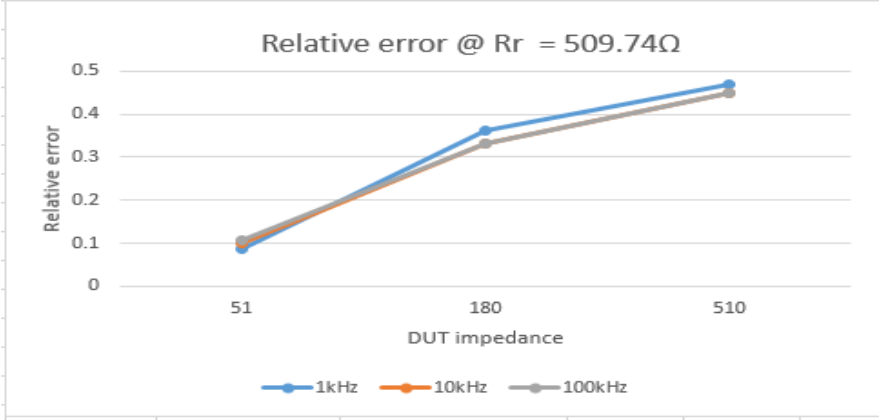

Figure 8: Relative error with low resistive DUT

Accuracy of resistive DUT measurements started to degrade at resistance greater than 10kΩ. This is worth investigating and we found that switches in the multiplexer exhibited stray capacitance of 200-400pF each, a crucial piece of information that we did not foresee in our initial design. With these capacitances, the overall slew rate of auto-balancing bridge was reduced and this attenuated signals at high frequencies (such observations were seen at frequencies of 10kHz and 100kHz). This was worsened if a branch with a higher resistance was selected, which would lead to a larger RC time constant.

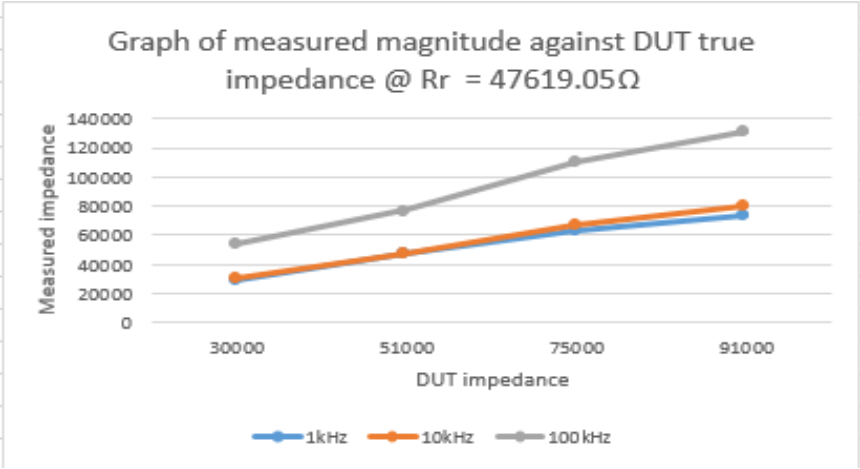

Figure 9: Measured magnitude against true impedance with range resistor = 48525 ohms. Note that value is only accurate at 1kHz

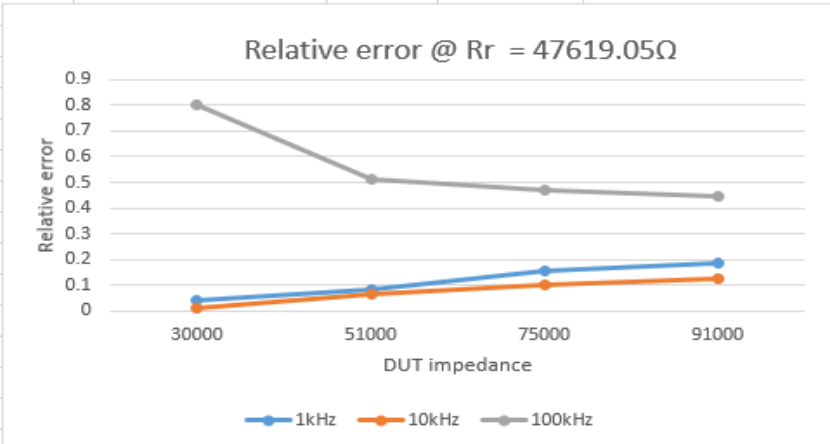

Figure 10: Relative error at range resistance = 47619.05 ohms

Phase delay was now significant as resistive DUT approaches 100kΩ in magnitude. This is because microcontroller would have to switch to higher resistance branch for magnitude computation, where stray capacitance dominates in this region

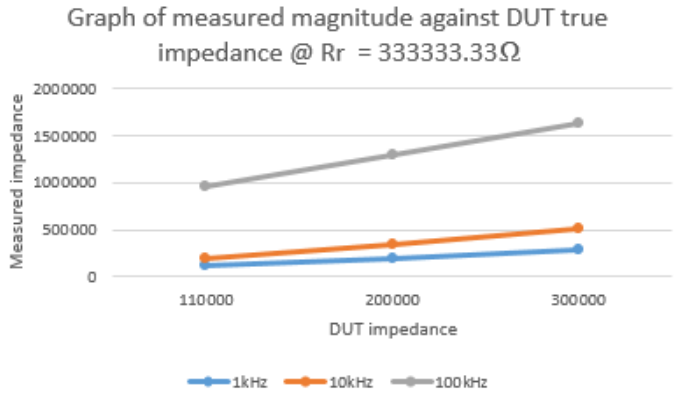

Figure 11: Magnitude of measured magnitude against true impedance with range resistor = 333333.33ohms. Note that value is only accurate at 1kHz.

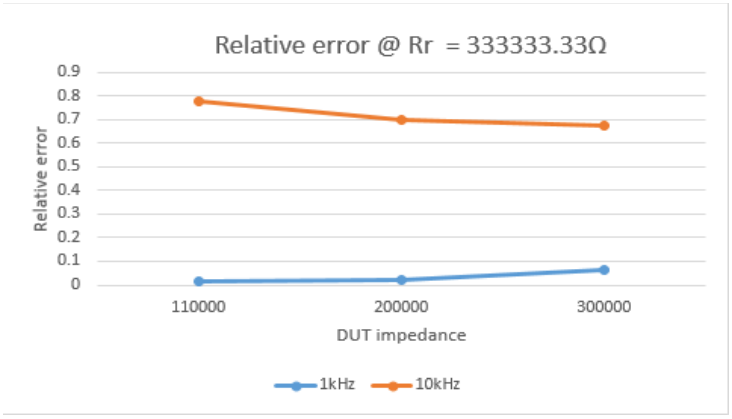

Figure 12: Relative error at range resistance = 333333.33ohms. 100kHz relative error not shown as this value has exceeded 1

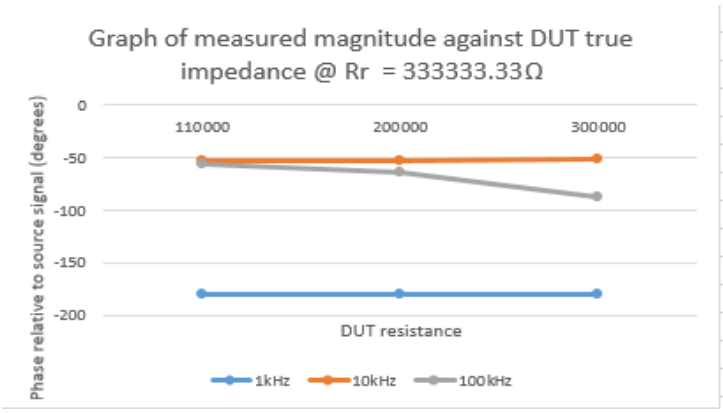

Figure 13:Phase of output signal relative to source signal with DUT resistance as shown.

Group Member Role Contributions

Yaohua Zhang

Responsible for:

- Ordering of components
- Designing of PCB
- Testing of individual stages
- Soldering of PCB

Liang Jun Wong

Responsible for:

- Writing and debugging microcontroller code
- Designing of PCB
- Testing of individual stages
- Soldering of PCB

Bill of Materials

Component	Price per unit	Quantity	Cost
MCP6292 Op-amp	£0.883	5	£4.415
74HCT14 SCHMIDT TRIGGER INVERTER	£0.47	1	£0.47
CD74HC4046AE PLL	£0.488	2	£0.976
CD4049UBE Hex Inverting Buffer	£0.405	2	£0.81
MPY634 4-quadrant Multiplier	£18.23	2	£36.46
74HC74 Dual D Flip-flop	£0.305	5	£1.525

20 PIN SIL VERTICAL SOCKET TIN	£1.44	2	£2.88
MCP1804T-5002I/OT Voltage Regulator	£0.52	2	£1.04
ADG1604BRUZ-REEL7 Multiplexer	£3.84	3	£11.52
New PCBs from PCBway	£4.468 (inclusive of delivery)	5	£22.34
			Total Cost (excluding new PCBs): £60.096
			Total Cost (including new PCBs): £82.436

Unfortunately, we exceeded the budget of £40. However, we reviewed each order seriously before making the payment. This is evidenced by the fact that the only components that did not make it to the final PCB were the 74HCT14 SCHMIDT TRIGGER INVERTER and CD74HC4046AE Phase-locked Loop, amounting to a tiny fraction of the total cost (£1.446).

We present 2 different total costs. The first cost only includes the cost of components that were ordered by us through EE-Stores (£60.096). In order to save cost, we borrowed components where possible. We borrowed the OLED screen from May Tang, AD9833 from Thomas Poskitt, sockets from Amine Halimi and various passive components from the EE-Lab. The second cost included the cost of new PCBs from an external vendor (PCBway), which we paid from our own pockets, as the 2nd PCB provided by the department had unresolvable issues (poorly drawn PCB tracks).