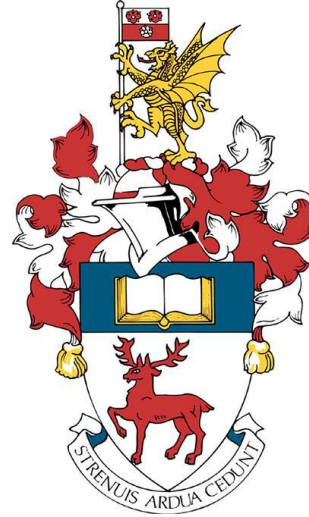


Electronics and Computer Science  
Faculty of Physical Sciences and Engineering  
University of Southampton

**UKESF** in collaboration with **University of Southampton**



**Authors:**

Eamonn Chislett-Trim  
Gavin Fish  
Leonardo Moreira  
Foivos Gaitantzis  
Adel Hedayat

# **Group Design Project 18: UKESF: Electronics Everywhere Companion Application**

**Project supervisor:** Professor Geoff Merrett

**Project client:** Stewart Edmondson, UKESF

**Second examiner:** Dr Mark Weal

A Group Design Project Report submitted for the Award of:  
*(MEng) Electrical and Electronic Engineering* and  
*(MEng) Computer Science*

**Date:** 15/01/2021

**Words:** 24696



## **Abstract** (Adel)

Electronics Everywhere Application is a project that aims to work alongside the UKESF/University of Southampton's outreach electronics kits to produce a cross-platform (iOS and Android) application that contains the tools for students to be able to interact with the Music Mixer board without highly technical laboratory equipment.

The Model View Controller architecture was adopted for use with Flutter to develop the application. The application has a practical page that has the oscilloscope and signal generator functionality. Lab instructions, interactive image and about pages were implemented for each board.

Feedback from A-level teachers states 93% agree or strongly agree that students would be capable of using the application. The application was tested to have an average margin of error below 1%. The application has implemented all the core specification and is ready to be used.

The client of the project has stated that the project was "innovative", "well planned" and "very organised" and "the group understood both the context and the design brief". The final thoughts, from the client, on the project were that it is "worthy of use" and "met all major goals".



# Table of Contents

<b>ABSTRACT (ADEL)</b> .....	<b>I</b>
<b>TABLE OF CONTENTS</b> .....	<b>III</b>
<b>ACKNOWLEDGEMENTS</b> .....	<b>VII</b>
<b>STATEMENT OF ORIGINALITY</b> .....	<b>IX</b>
<b>TABLE OF FIGURES</b> .....	<b>XI</b>
<b>ABBREVIATIONS AND DEFINITIONS</b> .....	<b>XV</b>
<b>1 INTRODUCTION (FOIVOS)</b> .....	<b>1</b>
<b>2 BACKGROUND RESEARCH</b> .....	<b>3</b>
2.1 KITS (GAVIN).....	3
2.1.1 <i>Music Mixer (Gavin)</i> .....	3
2.1.2 <i>Logic and Arithmetic (Gavin)</i> .....	5
2.2 LINEAR PULSE CODE MODULATION (LPCM) (FOIVOS) .....	6
2.2.1 <i>Sampling Rate and Resolution (Foivos)</i> .....	7
2.2.2 <i>PCM Conversion Process (Foivos)</i> .....	7
2.2.3 <i>Mobile Devices (Foivos)</i> .....	7
2.3 TOOLS (ADEL).....	8
2.3.1 <i>Native Application Development (Adel)</i> .....	8
2.3.2 <i>Cross-Platform Development (Adel)</i> .....	8
2.4 OTHER APPLICATIONS (LEONARDO) .....	9
2.4.1 <i>Dual-Channel Function Generator (Leonardo)</i> .....	9
2.4.2 <i>AUDio MEasurement System (Leonardo)</i> .....	9
2.5 STANDARDS OF DESIGN (EAMONN).....	10
2.5.1 <i>Model View Controller (Eamonn)</i> .....	10
2.5.2 <i>Version Control System (Eamonn)</i> .....	12
<b>3 SPECIFICATION (EAMONN)</b> .....	<b>13</b>
3.1.1 <i>Core Specification (Eamonn)</i> .....	13
3.1.2 <i>Stretch Goals (Eamonn)</i> .....	13
<b>4 DESIGN AND IMPLEMENTATION</b> .....	<b>15</b>
4.1 INITIAL DESIGN (EAMONN).....	15
4.1.1 <i>Model View Controller (Eamonn)</i> .....	15
4.2 HARDWARE IMPLEMENTATION (EAMONN).....	17
4.2.1 <i>Signal Generator (Eamonn)</i> .....	17
4.2.2 <i>Oscilloscope (Foivos)</i> .....	19
4.3 HOME SCREEN (LEONARDO) .....	21
4.3.1 <i>Page Components (Leonardo)</i> .....	21
4.3.2 <i>Initial Design and Implementation (Leonardo)</i> .....	21
4.3.3 <i>Design Changes and Considerations (Leonardo)</i> .....	21
4.3.4 <i>Final Design and Implementation (Leonardo)</i> .....	22
4.4 ABOUT PAGE (LEONARDO) .....	25
4.4.1 <i>Page Components (Leonardo)</i> .....	25
4.4.2 <i>Design Decisions (Leonardo)</i> .....	25
4.4.3 <i>Implementation (Leonardo)</i> .....	25
4.4.4 <i>Music Mixer Screen (Leonardo)</i> .....	26
4.4.5 <i>Logic and Arithmetic Screen (Leonardo)</i> .....	26
4.5 INTERACTIVE IMAGE PAGE (ADEL) .....	27
4.5.1 <i>Page Content (Adel)</i> .....	27
4.5.2 <i>Design Changes (Adel)</i> .....	28
4.5.3 <i>Final Design and Implementation (Adel)</i> .....	28

4.6	EXERCISES PAGES (LEONARDO) .....	32
4.6.1	Page Components (Leonardo) .....	32
4.6.2	Initial Design and Implementation (Leonardo) .....	32
4.6.3	Design Changes and Considerations (Leonardo) .....	32
4.6.4	Final Design and Implementation (Leonardo) .....	33
4.6.5	Logic and Arithmetic Screen (Leonardo).....	38
4.7	SETTINGS PAGE (LEONARDO).....	40
4.7.1	Page Components and Design Considerations (Leonardo).....	40
4.7.2	Final Design and Implementation (Leonardo) .....	40
4.8	PRACTICAL PAGE (GAVIN) .....	40
4.8.1	Design Considerations (Gavin) .....	40
4.8.2	Page Layout and Overview Design (Gavin) .....	41
4.8.3	Sidebar and Parameter Menu (Gavin).....	42
4.8.4	Graphing (Gavin) .....	46
4.8.5	Save/Load Slots (Foivos) .....	50
4.9	APPLICATION NAVIGATION (LEONARDO) .....	52
4.10	INTEGRATION (EAMONN).....	53
<b>5</b>	<b>TESTING (EAMONN).....</b>	<b>55</b>
5.1	METHODOLOGY (EAMONN) .....	55
5.2	ANDROID RESULTS (EAMONN) .....	56
5.2.1	Samsung S8 Using an Auxiliary Cable (Eamonn) .....	56
5.2.2	OnePlus One Using an Auxiliary Cable (Eamonn) .....	57
5.2.3	Samsung S8 Using Other Connections (Eamonn) .....	58
5.3	iOS RESULTS (EAMONN) .....	58
5.4	OTHER NOTABLE RESULTS (EAMONN).....	60
5.4.1	Signal Generation (Eamonn).....	60
5.4.2	Microphone Detection and Processing (Eamonn).....	62
5.4.3	Oscilloscope (Eamonn).....	62
5.5	A-LEVEL TEACHERS FEEDBACK (EAMONN) .....	63
5.5.1	First Survey (Eamonn).....	63
5.5.2	Second Survey (Eamonn).....	63
5.5.3	Third Survey (Eamonn).....	64
<b>6</b>	<b>FINAL OUTPUTS (GAVIN).....</b>	<b>69</b>
6.1	APPLICATION (EAMONN).....	69
6.2	HANDOVER RESOURCES (GAVIN).....	69
6.2.1	Handover Document (Gavin).....	69
6.2.2	Instructional Video (Gavin) .....	69
6.2.3	Written Instructions (Gavin).....	69
6.2.4	Music Mixer Laboratory Experiment Instructions (Gavin) .....	69
6.2.5	Stock Images (Gavin).....	70
6.3	SURVEYS (EAMONN) .....	70
6.4	TESTING DOCUMENTS (EAMONN).....	70
<b>7</b>	<b>PROJECT MANAGEMENT (EAMONN).....</b>	<b>71</b>
7.1	DIVISION OF RESPONSIBILITIES AND PROJECT PLANNING (EAMONN).....	71
7.1.1	Initial Plan (Eamonn) .....	71
7.1.2	Initial Division of Roles (Eamonn) .....	73
7.1.3	Actual Execution (Eamonn) .....	73
7.1.4	Actual Division of Roles (Eamonn).....	75
7.2	ETHICAL APPROVAL AND PLANNING OF QUESTIONNAIRES (EAMONN) .....	77
7.2.1	Initial Rationale and Planning (Eamonn).....	77
7.2.2	Evaluation of Usefulness (Eamonn).....	77
7.3	PLANNING AND TOOLS (EAMONN) .....	78
7.3.1	Version Control System (Eamonn).....	78
7.4	RISK MANAGEMENT (EAMONN) .....	79

7.4.1	<i>Initial Risk Assessment (Eamonn)</i> .....	79
7.4.2	<i>Actual Risks Mitigated (Eamonn)</i> .....	79
7.5	STRENGTHS AND WEAKNESSES MATRIX (EAMONN) .....	81
7.5.1	<i>Initial Strengths and Weaknesses (Eamonn)</i> .....	81
7.5.2	<i>Strengths and Weaknesses After Project (Eamonn)</i> .....	81
7.6	ACHIEVEMENTS AND RESULTS (EAMONN) .....	82
<b>8</b>	<b>CONCLUSION</b> .....	<b>85</b>
8.1	SPECIFICATION (EAMONN) .....	85
8.2	CONCLUDING DISCUSSION (FOIVOS).....	86
<b>9</b>	<b>FURTHER WORK (GAVIN)</b> .....	<b>89</b>
9.1	PRACTICAL PAGE TUTORIAL (GAVIN).....	89
9.2	LINKS FROM THE EXPERIMENT INSTRUCTIONS TO THE PRACTICAL PAGE (GAVIN) .....	90
9.3	INTERACTIVE 3D CAD MODEL (GAVIN).....	91
9.4	CROSS-PLATFORM SUPPORT FOR WEB BROWSERS (GAVIN) .....	91
9.5	LOGIC AND ARITHMETIC HARDWARE SUPPORT (GAVIN).....	91
9.6	SLIDE OUT GESTURE FOR SETTINGS BAR (GAVIN).....	92
9.7	PINCH TO ZOOM FOR THE PRACTICAL PAGE (GAVIN).....	93
9.8	TRIGGERING BASED ON MOVABLE TRIGGER POINT (GAVIN).....	93
9.9	SUPPORT FOR MICROPHONE INPUTS TO PHONE DIRECTLY FROM THE MUSIC MIXER BOARD (GAVIN) .....	94
<b>10</b>	<b>REFERENCES</b> .....	<b>97</b>
<b>11</b>	<b>APPENDIX</b> .....	<b>99</b>
	A ORIGINAL PROJECT BRIEF .....	100
	B SPECIFICATION .....	101
	C GITHUB BRANCH HISTORY .....	104
	D RISK ASSESSMENT .....	106
	E CLIENT FEEDBACK .....	108
	<i>E.i. Client Response</i> .....	108
	<i>E.ii. Initial E-mail</i> .....	109
	F EDITED LABORATORY NOTES .....	110
	G STOCK PHOTO HIGHLIGHTS.....	116
	H APPLICATION INSTRUCTIONS .....	118
	I HANDOVER DOCUMENT.....	127
	<i>I.i. Introduction and Purpose of Document</i> .....	127
	<i>I.ii. List of Features</i> .....	128
	<i>I.iii. Known Bugs</i> .....	131
	<i>I.iv. Software Information and Suggested Work</i> .....	133
	<i>I.v. Hardware Information and Suggested Work</i> .....	149
	<i>I.vi. Suggested Changes for Supporting Content</i> .....	154
	<i>I.vii. Additional Information</i> .....	155
	<i>I.viii. Project Structure</i> .....	157





# Acknowledgements

Thanks to the anonymous A-level physics teachers who gave their time to the surveys throughout the project.

Thanks to Dr Daniel Spencer, Dr Alex Weddell, and Dr Mark Weal for their feedback and continued support throughout the project.

Thanks to Professor Geoff Merrett from the University of Southampton and Stewart Edmondson from the UKESF, whom without their support, this project would not have been as successful.



# Statement of Originality

We have read and understood the ECS Academic Integrity information and the University's Academic Integrity Guidance for Students.

We are aware that failure to act following the Regulations Governing Academic Integrity may lead to the imposition of penalties which, for the most severe cases, may include termination of the program.

We consent to the University copying and distributing any or all of our work in any form and using third parties (who may be based outside the EU/EEA) to verify whether our work contains plagiarised material and quality assurance purposes.

We have acknowledged all sources and identified any content taken from elsewhere.

The use of survey responses from A-level Teachers was approved through the University of Southampton's ERGO board, and the scope is limited to use within the context of this project.

We have used the following open-source, free-to-use libraries for our implementation of the project:

- Shared Preferences for Flutter from GitHub [1]
- URL Launcher for Flutter from GitHub [2]
- Permission Handler for Flutter from GitHub [3]
- Flutter Launcher Icons for Flutter from GitHub [4]

We have adapted the library Mic Stream for Flutter from GitHub [5] to work with iOS and Android on this project.

We did all the work ourselves and have not helped anyone else.

The material in the report is genuine, and we have included all our data and code.

We have not submitted any part of this work for another assessment.

Our work did not involve human participants, their cells or data, or animals.



# Table of Figures

Figure 1: Music Mixer kit (sourced from [10]).....	3
Figure 2: Wave interference section of the Music Mixer board (sourced from [10]) .....	4
Figure 3: Capacitor discharge section of the Music Mixer board (sourced from [10]) .....	4
Figure 4: Plank's constant section of the Music Mixer board (sourced from [10]) .....	5
Figure 5: Logic and Arithmetic kit (sourced from [11]) .....	5
Figure 6: Logic section of the Logic and Arithmetic board (edited from [12]) .....	6
Figure 7: Arithmetic section of the Logic and Arithmetic board (edited from [12]) .....	6
Figure 8: Snippet of the Dual Channel Function Generator application (sourced from [21]).....	9
Figure 9: AUDio MEAsurement System in the oscilloscope tab (sourced from [22]) .....	10
Figure 10: MVC flow diagram (recreated from [23]) .....	10
Figure 11: The MVC extended to include a hardware interface .....	11
Figure 12: The original application layout designed on Figma at the start of the project.....	16
Figure 13: The Android implementation of the signal generator to fill the buffers .....	18
Figure 14: The iOS implementation of the signal generator to fill the buffers .....	19
Figure 15: Configuration of the mic_stream library for Android oscilloscope implementation .....	20
Figure 16: The original designs for the home screen showing (Left) the about page (Middle) the home page (Right) settings page .....	21
Figure 17: Final design of the home screen .....	22
Figure 18: AppBar widget showing the separate parts .....	23
Figure 19: Container widgets found on the home screen for the kits.....	23
Figure 20: Examples of the styling from the UKESF website (sourced [27]) .....	24
Figure 21: PCB trace widget taken from (Left) the UKESF website [27] (Right) the application .....	24
Figure 22: About page for (Left) the Music Mixer (Right) the Logic and Arithmetic.....	25
Figure 23: Button widget from (Left) the UKESF website [27] (Right) the application .....	26
Figure 24: Information widget from (Left) the UKESF website [27] (Right) the application .....	26
Figure 25: Initial design of the interactive pages for (Left) the Music Mixer (Middle) the Logic and Arithmetic (Right) the additional information .....	27
Figure 26: Final design of the tooltip for the interactive image page .....	29
Figure 27: Tooltip animation showing the passage of time from left-to-right.....	30
Figure 28: RotatingImage widget for NAND Gate additional information page (cycle goes top-left, top-right, bottom-left, bottom-right) .....	31
Figure 29: Initial Design for the exercise page .....	32
Figure 30: Final design for the exercise pages.....	33
Figure 31: The exercises containers (Left) static (Right) in a transition .....	34
Figure 32: Code for updating the padding for each of the lab containers .....	34
Figure 33: Training handbook download after pressing the button on the Music Mixer lab page.....	35
Figure 34: The four introduction pages for each section of the Music Mixer lab.....	36

Figure 35: The four step-by-step pages for each section of the Music Mixer lab .....	37
Figure 36: Exercises page for the Logic and Arithmetic screen .....	38
Figure 37: An example of the problem exercises for Logic and Arithmetic (Problem 2) .....	39
Figure 38: Image popup on the problem exercises for the Logic and Arithmetic .....	39
Figure 39: Final design of the settings page (Left) with dark mode enabled (Right) without dark mode enabled .....	40
Figure 40: practical page with annotations of the three main sections .....	41
Figure 41: Parameter menu and sidebar on the practical page (Left) expanded (Right) collapsed .....	42
Figure 42: Signal generator parameter menu showing (Left) top section; bottom section with (Middle-Left) left selected (Middle-Right) both selected (Right) right selected .....	43
Figure 43: Outputting box with (Left) both signals turned on (Middle) left signal turned on (Right) none turned on .....	43
Figure 44: Wave parameter selection tabs while the “Left” channel is selected .....	44
Figure 45: The user interface for editing (Top Left) Wave Type (Top Right) Frequency (Bottom Left) Amplitude (Bottom Right) Phase .....	44
Figure 46: Text editing of the frequency for the left channel .....	44
Figure 47: Attempting to input an out-of-range value into the frequency text entry .....	45
Figure 48: Oscilloscope parameter menu .....	45
Figure 49: Combined parameter menu (symbols representing a vertical continuation between sections) ..	46
Figure 50: Signal generator mode showing (Top) two graphs (Bottom) one graph .....	47
Figure 51: Combined mode showing (Top) three graphs (Middle) two graphs (Bottom) one graph .....	47
Figure 52: Text parameters displayed at the top left of the graph for (Left) a single signal on (Middle) a single signal off (Right) multiple signals (one on, one off) .....	48
Figure 53: Code for finding the scale between two frequencies of signal generation .....	49
Figure 54: A 20 Hz sine wave is shown on the combined screen using (Left) a total time of 12 s (Right) a total time of 0.12 ms .....	49
Figure 55: A demonstration of the triggering working for a signal generated wave .....	50
Figure 56: Save/load parameters popup with middle slot free .....	50
Figure 57: Confirmation dialogue for saving to a slot .....	51
Figure 58: Confirmation dialogue for deleting a slot .....	51
Figure 59: Information popup for the save/load slots .....	51
Figure 60: Go home popup for kit pages .....	52
Figure 61: Exit button on the practical page .....	52
Figure 62: Testing configuration using a Samsung S8 .....	55
Figure 63: Graph showing the target frequency compared to the recorded frequency using the auxiliary cable on a Samsung S8 .....	56
Figure 64: Graph showing the target phase compared to the recorded phase using the auxiliary cable on a Samsung S8 .....	56
Figure 65: Graphing showing recorded amplitudes for different phone and slider volumes on a Samsung S8 .....	57

Figure 66: Graphing showing recorded amplitudes for different phone and slider volumes on a OnePlus One .....	57
Figure 67: Graph showing the target frequency compared to the recorded frequency for using the lightning to auxiliary cable on an iPhone XS .....	58
Figure 68: Graph showing the target phase compared to the recorded phase using the lightning to auxiliary cable on an iPhone XS .....	59
Figure 69: Graph showing recorded amplitudes for different phone and slider options on an iPhone XS ..	59
Figure 70: Triangle wave being generated by a OnePlus One(Top) correctly at 2500 Hz (Bottom) incorrectly at 250 Hz.....	60
Figure 71: Triangle wave being generated by a Samsung S8 (Top) correctly at 2500 Hz (Bottom) incorrectly at 5000 Hz.....	61
Figure 72: Constructive wave of 1260 Hz sine wave with a 440 Hz sine wave (Top) on the application (Bottom) on the laboratory oscilloscope.....	62
Figure 73: Constructive wave of 440 Hz sine wave with a 3430 Hz square wave with a 180° phase offset (Top) on the application (Bottom) on the laboratory oscilloscope .....	63
Figure 74: First survey responses .....	65
Figure 75: Second survey results .....	66
Figure 76: Third survey results .....	67
Figure 77: Initial Gantt chart with the sprints labelled .....	72
Figure 78: Actual Gantt chart execution .....	74
Figure 79: Overview of the GitHub branch history for the project, taken with GMaster [32].....	78
Figure 80: Example of how the risk assessment was created, using lockdown due to Covid-19 as an example .....	79
Figure 81: Initial strengths and weakness matrix.....	81
Figure 82: Final strengths and weaknesses matrix.....	82
Figure 83: Mock-up design of practical page tutorial.....	89
Figure 84: Mock-up diagram of the transitions to and from the instructions and practical page.....	90
Figure 85: Placeholder button to launch to the practical page from the instructions .....	91
Figure 86: Practical page parameter menu, showing the collapse button (Left) expanded (Right) collapsed .....	92
Figure 87: Use case of the board redesign with TRRS connection to three phones with splitters .....	94
Figure 88: Use case of the board redesign with TRRS connection to a single phone that is used as both oscilloscope and signal generator .....	95





# Abbreviations and Definitions

## V.I Abbreviations

<b>UKESF</b>	United Kingdom Electronics Skills Foundation
<b>GUI</b>	Graphical User Interface
<b>PCB</b>	Printed Circuit Board
<b>LED</b>	Light Emitting Diode
<b>LDR</b>	Light Dependent Resistor
<b>TRS</b>	Tip Ring Sleeve, referring to the auxiliary 3-pole connection
<b>TRRS</b>	Tip Ring Ring Sleeve, referring to the auxiliary 4-pole connection
<b>Op-amp</b>	A shorthand for “operational amplifier”
<b>MVC</b>	Model View Controller
<b>PCM</b>	Pulse Code Modulation
<b>SDK</b>	Software Development Kit

## V.II Definitions

<b>Backend</b>	Referring to the coding implementation that corresponds to functionality, and not GUI elements
<b>Kits</b>	Referring to the Electronic Engineering Kits that have been worked on by the University of Southampton’s ECS department
<b>Board</b>	Referring to the complete PCB board part of the specific Electronic Engineering Kit
<b>Widget</b>	Referring to a Flutter widget, which is a piece of code that relates a singular GUI object
<b>practical page</b>	Referring to the Electronics Everywhere Application page to interface with the Music Mixer board



# 1 Introduction (Foivos)

“In the UK, the Electronics sector is big, valuable and growing; however, the demand for capable employable graduates is currently outstripping supply” [6].

— United Kingdom Electronics Skills Foundation

As of spring 2018, the UK had the 6<sup>th</sup> largest Electronics industry globally with a £98 billion annual turnover contributing to 6% of its GDP and providing over 1,000,000 related jobs [7]. The UK Electronics Skills Foundation (UKESF) is an educational charity focusing on ensuring that high-school children are aware of electronics and its opportunities. It operates alongside leading universities, and major corporations, working closely on practical initiatives that tackle the skills shortage.

Electronics Everywhere is a subset of the UKESF (in collaboration with the University of Southampton) that makes electronics more engaging by producing specifically designed circuit boards to teach core electronics concepts to A-level students undertaking computer science and physics [8] (these are discussed further in 2.1). The UKESF distribute these boards to hundreds of schools (thousands of users) around the country (as stated in appendix A). Since most A-level physics students are kinaesthetic learners, the kits have been vital in showing young people how engaging electronics can be. Unfortunately, specific electronic lab equipment may be inaccessible or limited such as a signal generator and an oscilloscope that are mandatory for the students to interface with the kits.

This project aims to design, implement, and test a cross-platform application that provides the functionality of a signal generator and an oscilloscope on both iOS and Android. This application is targeted at being used alongside the kits already distributed. The client, Stewart Edmondson, who is the CEO of the UKESF, explicitly asked for a polished application with fewer bugs than one with more features. Another critical requirement included having strong branding throughout the application. Other applications previously recommended by the UKESF have not been created for A-level students' use due to their complexity in design. This project looks to simplify the design to be tailored to A-level physics students and provide a full experience working with the physical kits provided.

The societal goal is to positively impact A-level physics students' engagement, potentially inspiring more students to take up electronics at university. Cost of equipment for A-level physics classes will no longer be an issue, as the application will be provided for free with the UKESF Electronics Everywhere kits.



## 2 Background Research

### 2.1 Kits (Gavin)

The UKESF and academics at the University of Southampton’s ECS department have collaborated to design, produce and distribute some educational “Electronic Engineering Kits” [9]. These kits are aimed at introducing A-level students to electronic circuits and concepts. The two main kits explored in this project are the Music Mixer, and the Logic and Arithmetic kit.

#### 2.1.1 Music Mixer (Gavin)

The Music Mixer board demonstrates three electronics concepts on a single board: wave interference, capacitor-discharge, and Plank’s constant [10]. Figure 1 shows that the Music Mixer kit contains the Music Mixer board, two TRS auxiliary cables, an “aeroplane” auxiliary splitter, two LDRs and ten pin jumpers.

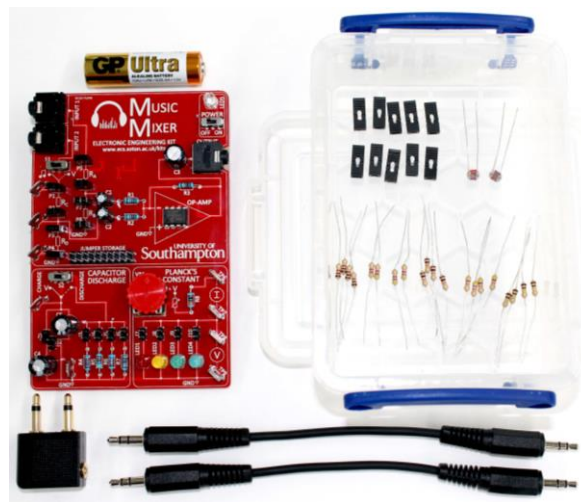


Figure 1: Music Mixer kit (sourced from [10])

##### 2.1.1.1 Existing Resources (Gavin)

There are supporting resources for the Music Mixer kit available online [10]. A student-friendly user guide is included in the Music Mixer kit, which gives an overview of the three sections of the board. A similar guide aimed at teachers contains more information about the circuitry of the board and how it might be used that can be found online.

A training handbook contains instructions for multiple laboratory experiments with the Music Mixer board. The training handbook references applications that can be used alongside the board if access to laboratory oscilloscopes or signal generators is limited. Section 2.2 “Potential Dividers (AC)” of the training handbook, in particular, could benefit from the application developed in this project.

Some short informational videos run through experiments, electronic circuits and concepts found on the Music Mixer board. An in-depth online training video walks through how to use some third-party computer software to interface with the Music Mixer kit. This training video could also benefit from the application developed in this project.

### 2.1.1.2 Music Mixer Section (Gavin)

The Music Mixer section is the main section of the board and is shown in Figure 2. This section demonstrates the concepts of potential dividers and summation of signals via an op-amp. This is the part of the board that the application will be targeting primarily.



Figure 2: Wave interference section of the Music Mixer board (sourced from [10])

The input and output ports wiring are fundamental to planning the interface between this board and devices running the Electronics Everywhere Application. There are two mono auxiliary inputs to the board and a mono output.

### 2.1.1.3 Capacitor-Discharge Section (Gavin)

The capacitor-discharge section demonstrates the charging and discharging of capacitors by including two capacitors, four different sized resistors, and jumpers to reconfigure the circuit quickly. Either both of the capacitors can be connected, (in parallel or series) or only one connected. The resistors are set up in parallel and can either be connected or disconnected with a jumper. The switch S2 is used to charge or discharge the capacitors through the resistor configuration.

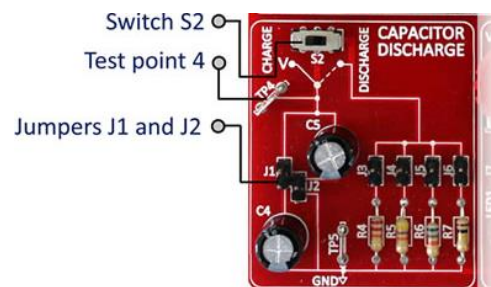


Figure 3: Capacitor discharge section of the Music Mixer board (sourced from [10])

### 2.1.1.4 Plank's Constant Section (Gavin)

This section of the board can be used to estimate Plank's constant by measuring the voltage and current characteristics of a range of different coloured LEDs [10]. The equipment used to run experiments with this section is a voltmeter and ammeter. These features are outside of the scope of this project.

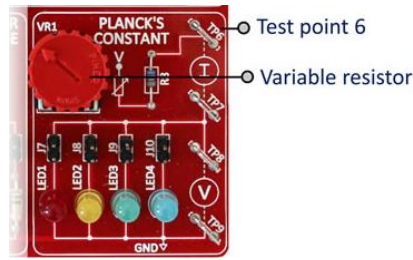


Figure 4: Plank's constant section of the Music Mixer board (sourced from [10])

### 2.1.2 Logic and Arithmetic (Gavin)

The Logic and Arithmetic board demonstrates electronic concepts such as combinatorial logic gates and binary number operations aimed at A-level computer science students [11]. Unlike the Music Mixer, this standalone board that does not require external equipment. The board has two sections: the combinatorial logic section, and the arithmetic section. An image of the board is shown in Figure 5.

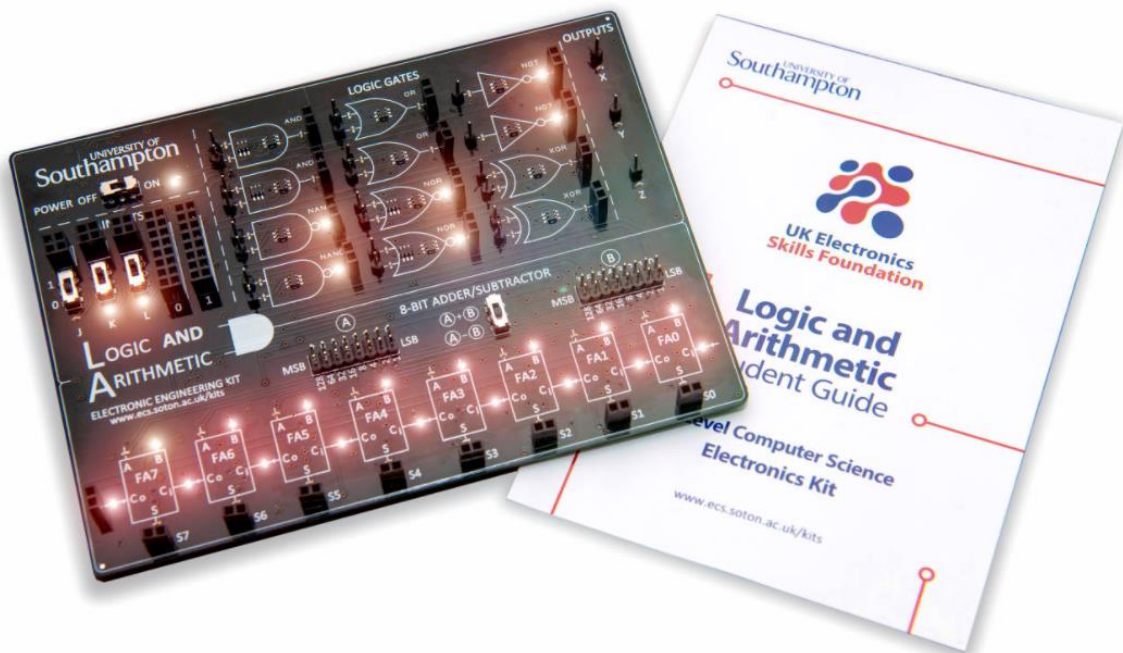


Figure 5: Logic and Arithmetic kit (sourced from [11])

#### 2.1.2.1 Existing Resources (Gavin)

There are supporting resources for the Logic and Arithmetic board available online [11]. A student-friendly user guide is included in the box that gives an overview of the two sections of the board. A similar guide aimed at teachers contains more information about the circuitry of the board and how it might be used. Some short informational videos run through some logical problems that can be run on the board. A training handbook contains instructions for multiple laboratory experiments that can be run on the two sections of the Logic and Arithmetic board. There is also a logic problem document that contains ten exercises that can run using the logic part of the board. There is a training video that walks through the experiments in the training handbook.

### 2.1.2.2 Logic Section (Gavin)

The logic section contains various combinatorial gates, input logic level switches, static logic levels and output LEDs to teach boolean operations and combinatorial logic. The logic section of the board is shown in Figure 6.

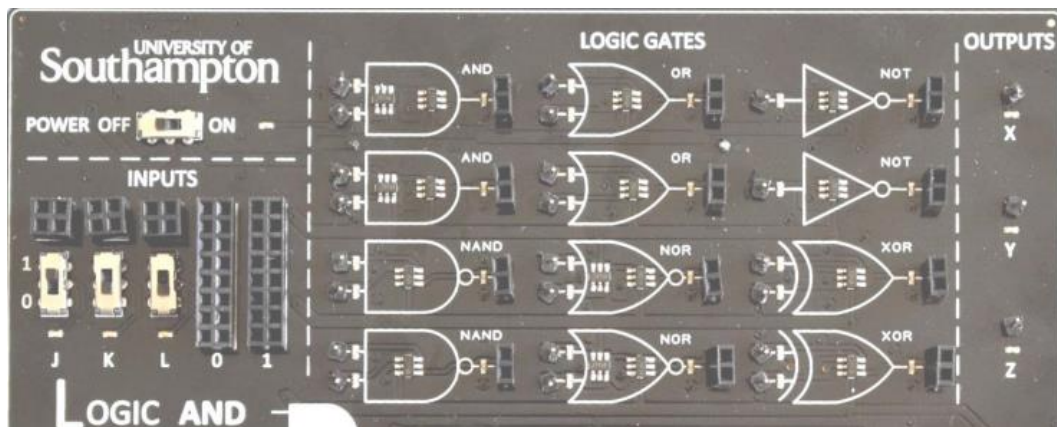


Figure 6: Logic section of the Logic and Arithmetic board (edited from [12])

Users can create combinatorial circuits using jumper cables to join from the input logic levels to numerous logic gates and then to the outputs. Many configurations of circuits can be realised, which gives flexibility for different exercises to be run. There are LEDs at every stage of this circuit, which makes visualising the logic levels simple.

### 2.1.2.3 Arithmetic Section (Gavin)

The arithmetic section comprises of eight full-adders, a switch to configure the full-adders into an adder or subtractor, and jumpers that can be used to control the inputs. This section can be used to aid in teaching binary number systems. The arithmetic section of the board is shown in Figure 7.

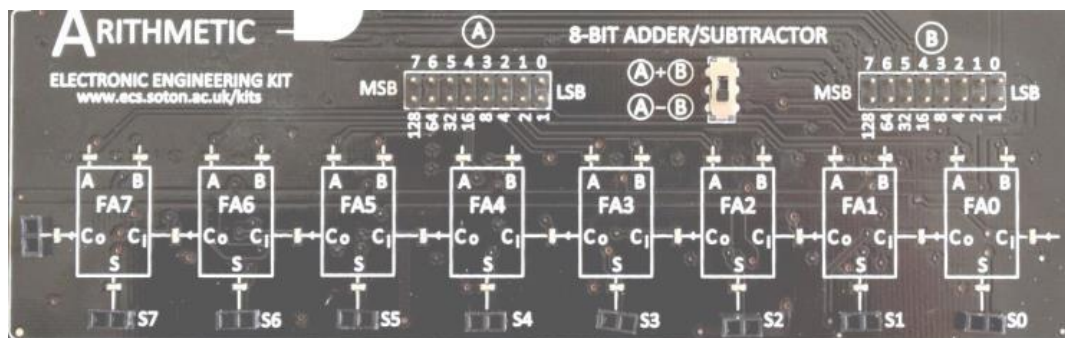


Figure 7: Arithmetic section of the Logic and Arithmetic board (edited from [12])

Users can add or subtract up to 8-bit binary numbers by configuring jumpers to configure the numbers input and toggling the adder/subtractor switch. There are LEDs at every input and output of the full-adders, which makes visualising the numbers simple.

## 2.2 Linear Pulse Code Modulation (LPCM) (Foivos)

Pulse Code Modulation (PCM) is used to convert a continuous analogue signal into a discrete-time digital signal. Each waveform sample is stored as a discrete integer value (up to a specific bit length) with no compression.



### 2.2.1 Sampling Rate and Resolution (Foivos)

There is a trade-off between signal quality and storage space required according to two key parameters: the sample rate and the resolution. The sample rate is the number of samples recorded per second, in Hz. The resolution is the number of bits of information in each sample. On most mobile devices, audio input is sampled at 44.1kHz with points that are 16-bits in size (resolution) [13].

### 2.2.2 PCM Conversion Process (Foivos)

The conversion process follows three key steps: sampling, quantisation and encoding. The signal is first sampled by measuring the amplitude at discrete time intervals. According to the Nyquist theorem, a signal can be regenerated without losing any information if it is sampled at a rate that is twice the highest frequency of the input signal [14].

In quantisation, the analogue samples range is divided into different levels (depending on the resolution) with the analogue samples approximated to their nearest quantisation values. Due to this approximation, the regenerated signal will always slightly differ to the original, often referred to as the quantisation error [15].

Finally, encoding ensures that the bandwidth is minimised by designating each quantised level to a binary code.

### 2.2.3 Mobile Devices (Foivos)

Android and iOS devices support Linear Pulse Code Modulation (LPCM), a PCM where quantisation levels are linearly uniform and incremental. The value at each sample is determined using equation 1 below, for a sine wave of given amplitude  $A$  and sample with index  $N$ .

$$F(N) = A \sin(\Theta(N)) \quad 1$$

The value of  $\Theta$  is determined using equation 2 below where  $F$  is the wave frequency,  $R$  is the sample rate, and  $N$  indicates the index of the sample.

$$\Theta(N) = 2\pi \frac{FR}{N} \quad 2$$

These raw audio samples are placed into output buffers. The inverse operation to sound generation.

#### 2.2.3.1 Android (Eamonn)

Several Android libraries handle the input and output for audio. These different libraries contain different levels of abstraction for the coder. The library that allows for LPCM output to be used is called AudioTrack [16]. Using this library, an output buffer can be filled with a set length of values for LPCM. This allows custom sounds to be synthesised for audio output. AudioRecord is used for LPCM audio input, allowing the display of the buffer as 16-bit values [17].

#### 2.2.3.2 iOS (Foivos)

On iOS, the AudioUnit component is the lowest level library for audio generation and processing. Additionally, it provides seven functions to handle mixing, format conversion, effects and input/output. For this project, only the remote input/output unit was used to obtain “low-latency access to individual incoming and outgoing audio sample values” [18].

## 2.3 Tools (Adel)

In the UK, the mobile phone operating system market is dominated by iOS and Android, accounting for 99.5% of mobile phone operating systems market share [19]. There is a need to develop a cross-platform application for both iOS and Android to reach these users. There are two ways to build a cross-platform application, developing native iOS and Android apps separately or using a cross-platform development tool.

### 2.3.1 Native Application Development (Adel)

In terms of the application, native development refers to developing an application for a single specific platform. This is achieved using native Software Development Kits (SDK) of the platform. These SDKs are written in particular languages: Objective-C or Swift for iOS, and Kotlin or Java for Android.

There are a few benefits to working natively such as direct access to the low-level libraries to work on the platform, better native performance, and more straightforward material design for the user interface. However, there are also downsides to native development, such as requiring each platform to have different teams and expertise. Additionally, each platform has different implementations that may not align due to the different teams working independently.

### 2.3.2 Cross-Platform Development (Adel)

Cross-platform development provides an alternative which aims to tackle the issues with developing separate native codes. A cross-platform development tool allows creating an application that can be deployed on multiple platforms from a single code base. This removes the need to split resources between various teams, leading to reduced development time, cost and faster deployment to market. Another benefit of cross-platform development is that it ensures the design of the application is uniform across platforms. A few frameworks are available, but the two most widely supported are React Native and Flutter [20].

#### 2.3.2.1 React Native (Adel)

React Native is a cross-platform framework developed by Facebook in 2015. In React Native, the application logic is written in JavaScript and is connected to the native SDK using a bridge which serves as an interpreter. This bridge comprises of two layers: The communication layer, and the React Native API. The first is the native modules responsible for communicating with the native SDK. These native modules are written in Objective-C for iOS and Java for Android. The second layer of the bridge is the React Native API written JavaScript. React Native has all of the advantages of cross-platform development but lacks performance speed due to the communication delays of the bridge.

#### 2.3.2.2 Flutter (Adel)

Flutter is a cross-platform framework developed by Google in 2017, which uses the dart programming language. In Flutter everything is made of widgets which are the building blocks of the application. A widget can be anything from an animation, image, text, gesture, to any other user interface component. These widgets are compiled to native ARM code ahead of time using native libraries. Flutter has all the advantages of cross-platform development with the added benefit of a long list of built-in Widgets, making development

relatively simple. The only drawback is the Flutter application occupies a larger amount of space due to the use of built-in widgets and not platform widgets.

As the project has a limited timeframe, native development would not be feasible. With the possibility of React Native communication delays hindering the real-time performance of oscilloscope, Flutter was chosen as the tool used in this project.

## 2.4 Other Applications (Leonardo)

The Dual-Channel Function Generator [21] and the AUDio Measurement System [22] are third-party software currently being used with the kits.

### 2.4.1 Dual-Channel Function Generator (Leonardo)

The Dual-Channel Function Generator developed by Keuwlsoft [21], can be downloaded from the Google Play Store and is the application currently recommended to be used with the kits. For example, it is used in the tutorial video on how to use the Music Mixer board on the website [10].

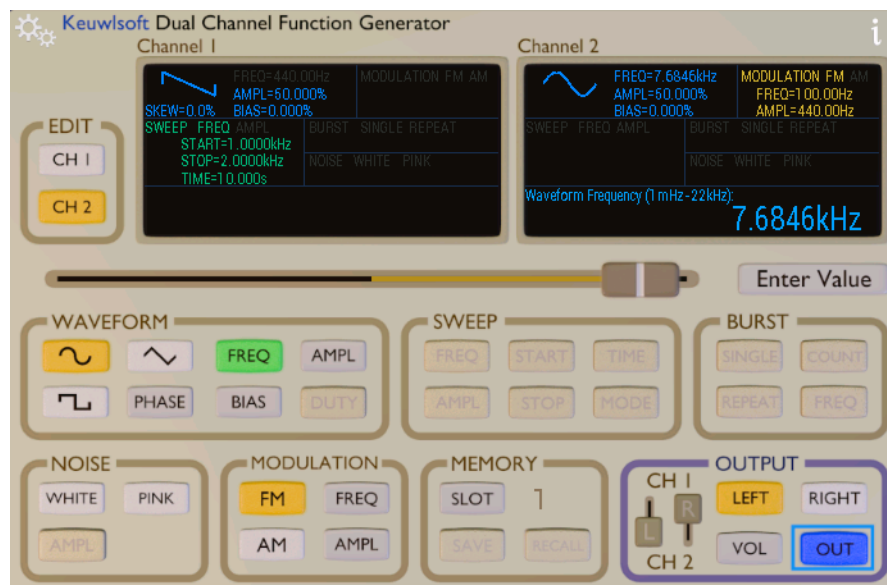


Figure 8: Snippet of the Dual Channel Function Generator application (sourced from [21])

As this application is already used with the kits, many aspects were used to inspire the user interface of the project. Of note is the ability to change between three distinct waveforms, output to the left and right channel independently, and change the left and right channel settings during operation.

### 2.4.2 AUDio MEasurement System (Leonardo)

The AUDio Measurement System is a desktop software for audio measurement and can be downloaded from SourceForge [22]. It was also mentioned in the project specification and used in the training video to use the Music Mixer kit on the website. As a desktop software, only the functionality was used for inspiration, as the user interface was for a desktop, not a mobile, application. However, the input settings and graphing style influenced design decisions and implementation.

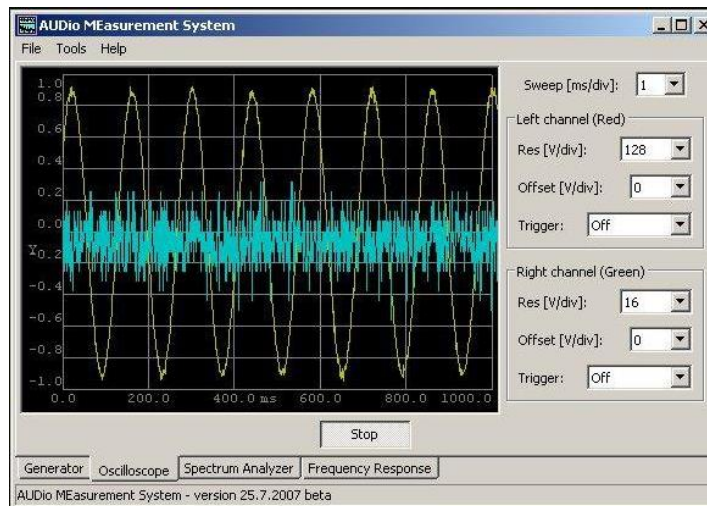


Figure 9: AUDIO MEasurement System in the oscilloscope tab (sourced from [22])

## 2.5 Standards of Design (Eamonn)

### 2.5.1 Model View Controller (Eamonn)

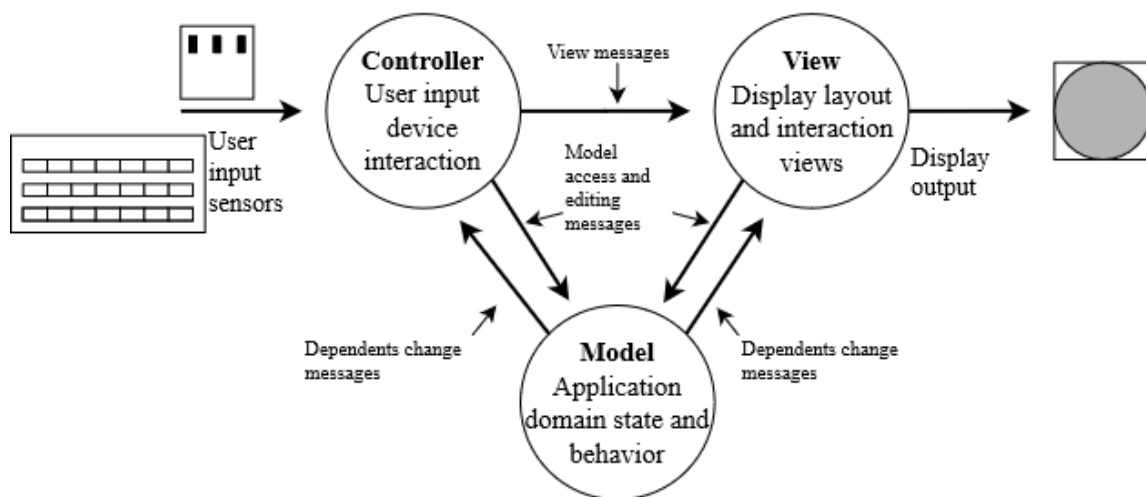


Figure 10: MVC flow diagram (recreated from [23])

The Model View Controller (MVC) standard was chosen to ensure that the coding of the project adhered to a consistent quality and structure level. The MVC standard is a method of segregating object-oriented programming to show the different ways of operation. The process of drawing an on-screen update is cyclical [23]. The user interacts with the device, causing a controller update. This is then sent to the model, generating change in the model. The difference in the model is updated to the view to display the user the most up-to-date information.

#### 2.5.1.1 Model (Eamonn)

The model is concerned with the memory storage and variables used in the operation of the class. The requirements are that the model contains the relevant variables that can be changed and interacted with via the controller, displayed by the view.

### 2.5.1.2 View (Eamonn)

The view is concerned with the display (user interface) of the model on screen for the user to see. The view contains all the information about the stylistic and visual aspects of the display. It often refers to the model variables using getters that allow the view to update when the model is updated.

### 2.5.1.3 Controller (Eamonn)

The controller is the functional part of the class. The controller handles the user inputs to update the model (which in turn updates the view). The controller contains all the methods and function calls that invoke setters in the model to interface with the user inputs correctly.

### 2.5.1.4 MVC Concerning this Project (Eamonn)

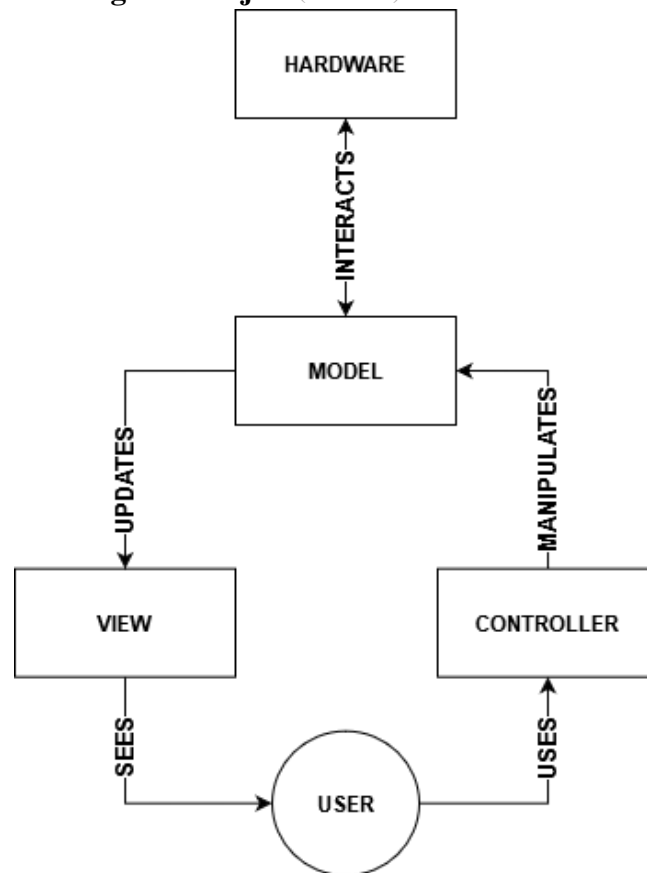


Figure 11: The MVC extended to include a hardware interface

This project has taken the MVC design and adapted it to work for the requirements. Flutter has much of this MVC design already embedded in the tools used. Since it is an object-oriented programming language that focuses on GUI aspects, the classes are already segregated to support MVC.

One area in which MVC does not explicitly have any support is handling hardware input/output variables. This consideration is made more difficult because of how Flutter handles native coding for hardware implementations, as it is written in a different coding language. As a result, the MVC design has been extended to include a hardware interface connected to the model. Hence, the model updates the hardware output, and the hardware input updates the model.

### **2.5.2 Version Control System** (Eamonn)

To ensure that the project could be correctly managed and full accountability in the coding practices, the University of Southampton GitLab (a Version Control System or VCS) was used. The benefits of a VCS are the complete history of code, the ability to branch and merge, and full traceability for the changes made [24].

## 3 Specification (Eamonn)

### 3.1.1 Core Specification (Eamonn)

A complete specification (appendix B) was created from a discussion with the client and the project supervisor building from the initial project brief (appendix A). The agreement in the specification was that the following list of core features was required for the application.

- *Visualisation of an input signal from the microphone*
  - *44.1kHz sample rate at 16-bit resolution*
  - *Ability to change the scale, relative phase to the generated signals and auto set*
- *Signal generator through the audio port*
  - *A maximum of 44.1kHz maximum voltages is 2V peak-to-peak at a 16-bit resolution at loads of 100-600 Ohms impedance*
  - *Two individual mono-channel signals on the left and right audio channel*
  - *Sine, square, triangle waves, music samples as default options*
  - *Ability to change the amplitude, relative phase, frequency*
  - *Can be individually turned off and on (if both are off, then music can be played)*
  - *To be directly visualisable alongside the input signal from the microphone*
- *Instruction sets for experiments*
  - *Targeted at students to follow (enabling simple lesson plans for teachers)*
  - *Covers all the material already available on the webpages*
  - *Provides contextual background knowledge and theory about the experiment*
- *User interface*
  - *Targeted primarily at students to use, but also with teachers in mind*
  - *Links to UKESF & University of Southampton tools*
  - *Branding and colour scheme matching UKESF and Electronics Everywhere*
  - *Focusing on functional features to support experiments, without too many additional to avoid an overwhelming interface*
  - *Save/load slots to retain specific configurations*

### 3.1.2 Stretch Goals (Eamonn)

Along with the original specification, a list of “stretch goals” were created. These tasks were not required for the project to be successful; however, it would enhance the experience of using the application. These included:

- *User interface*
  - *Pinch zooms and edits for signals*
  - *Interactive pictures of boards to give information about the operation*
  - *Addition instructional tasks to produce certain output signals (within tolerances) based on changing the circuit/signal configuration (providing the more adept students with a chance to explore the more difficult application of physics, i.e. using these two output signals, and using the board, produce a signal that is a mix of the first two)*
- *Logic analyser*
  - *Information on the board included within the application*
  - *Instructions for the experiments on the logic analyser found on the website*
  - *Produce a technical plan for implementing a logic analyser with the application (including hardware if required)*





## 4 Design and Implementation

### 4.1 Initial Design (Eamonn)

For many of the initial design ideas of the application, a collaborative interface design tool, Figma, was used. Once these initial designs were made, the team could more deeply think about the interactions within the application (Figure 12 shows these initial designs). This allowed for an iterative process of development. Once the team had developed a version, used it, and thought of feedback/changes made, it was then iterated upon for the final design. These reviews and iterations occurred in the sprints of the project (this will be discussed in 7).

#### 4.1.1 Model View Controller (Eamonn)

The Model View Controller (as explained in 2.5.1) was extremely influential in the project. The additional adaptation made to include the native side was instrumental in incorporating this software model to the project.

Initially, the decision was made to split every page and widget up into this file structure. This meant creating a model, a view, and a controller file for each implementation. This was quickly changed to only need an MVC structure for the more complicated pages or widgets in the project (for example, the Music Mixer practical page).

The benefit of handling the more complicated pages in this way meant that depreciated code, variables or functions were quickly seen, and cleared up. It also benefited from handling scope issues that might have caused some of the more complicated practical page features to be more challenging to implement. In this way, the benefit of the MVC structure was mainly in the pages where multiple team members worked in, and in combination with the VCS the benefits meant quick merging.



Figure 12: The original application layout designed on Figma at the start of the project

## 4.2 Hardware Implementation (Eamonn)

Part of the requirement of the application is to interface with the headphone port (or auxiliary port) to generate and receive signals. Native code was developed on both platforms individually to implement these features, with the implementation being specific to each platform. The Android backend is written in Java, and the iOS backend is developed in Objective-C. These native implementations, while being written in different languages, are functionally identical. The native backend implementation interfaces with the Flutter frontend to send data to and from the native side to Flutter.

### 4.2.1 Signal Generator (Eamonn)

The signal generator required producing three signals: a sine wave, a triangle wave, and a square wave. The respective formulae are included in Table 1, and these are the basis of the formulae used in the native implementations. The  $f$  symbol in the following formulae represents frequency, and the  $\theta$  symbol represents the relative phase (in radians) of the signals.

Table 1: Mathematical formulae for different periodic signals

<i>Sine</i>	$\sin(2\pi f + \theta)$
<i>Triangle</i>	$\frac{2}{\pi} \arcsin(\sin(2\pi f + \theta))$
<i>Square</i>	$\text{sign}(\sin(2\pi f + \theta))$

The combined implementation on the Flutter side of the project involved setting variables through a MethodChannel. This is essentially a method of creating functions called on the native application from the shared codebase. Once the required variables were set, and the native code had been implemented, it became a simple case of aligning iOS and Android functions.

#### 4.2.1.1 Android (Eamonn)

The signal generation implementation on Android involved using the built-in Android libraries that allow interfacing with the audio port to use generated tones. The Android implementation uses a thread that repeatedly fills the output buffer. These variables can be updated using a MethodChannel to regenerate the array which fills the output buffer. Using the native AudioTrack library, the audio output is set up as a stereo music stream, with 16-bit PCM encoding.

The output buffers size is determined by the sample rate and the refresh time of the buffer. Since the tone could be changing rapidly when a user interacts with a slider, and the output of the device should reflect this, the refresh time was set to 0.1 seconds. The sample rate is fixed at the maximum sample rate of the device (which for most devices is 44100 Hz). This buffer is filled using the formulae described in Table 1, with the actual code shown in Figure 13.

The phase offset variables in the formulae are in radians, yet in the code are in degree. As a result, the conversion is made in this implementation. The amplitude value ranges from zero to one, and as a result, is a pre-scalar of the maximum output amplitude of the phone. Combining these factors ensures that the output of “tempCalculation”, which is later appended into the output buffer, represents the signal shown on the graphical user

interface and can be measured and tested with external equipment to match the expected output.

```
if(leftTurnedOn){
  switch(leftWaveType){
    case sine:
      tempCalculation =
        (short) (Math.sin(2 * Math.PI * i * leftFrequency / sampleRate
          - ((Math.PI/180) * leftPhaseOffset))
          * (0x7FFF * leftAmplitude));
      break;
    case triangle:
      tempCalculation = (short) ((2 / Math.PI)
        * Math.asin(Math.sin(2 * Math.PI * i * leftFrequency / sampleRate
          - ((Math.PI/180) * leftPhaseOffset))))
        * (0x7FFF * leftAmplitude));
      break;
    case square:
      tempCalculation =
        (short) (Math.signum(Math.sin(2 * Math.PI * i * leftFrequency / sampleRate
          - ((Math.PI/180) * leftPhaseOffset))))
        * (0x7FFF * leftAmplitude));
      break;
    default:
      tempCalculation = (short) 0;
  }
} else {
  tempCalculation = (short) 0;
}
```

Figure 13: The Android implementation of the signal generator to fill the buffers

#### 4.2.1.2 iOS (Foivos)

On iOS, a similar implementation to Android was designed using the built-in library, AudioUnit. The lowest-level library was used as the more abstracted libraries do not easily allow for synthesising audio for output. The variables used to generate the signal are updated through the Flutter side of the project using the same MethodChannel as Android.

The AudioUnit for audio output is initialised as a dual-channel 32-bit floating-point Linear PCM to generate separate tones between the left and right channel. This was chosen because the most optimal float format on iOS is 8-bits relating to the exponent, and 24-bits relating to the fractional component [25]. The 32-bit implementation showed efficiency throughout testing and was, therefore, not changed. Additionally, a sample rate of 44.1kHz was desired; however, that will depend on the iOS device and audio output hardware.

It is also worth mentioning that the AudioUnit for playback is initialised whenever the practical page is launched and is destroyed whenever it closes (i.e. scenarios where the notification bar is visible). Once the AudioUnit is created and initialised, it will only run

using `AudioOutputUnitStarts`. At this point, no parameters can be modified unless the `AudioUnit` is uninitialised.

Like Android, the iOS implementation uses a thread that repeatedly fills in the output buffers according to the client-side parameters. The size of the buffer is automatically determined, and each frame on the buffer must be filled in with signal values. The buffer is filled by the code in Figure 14 where the integer variable “WaveTypeLeft” (ranging from one to three) represents a sine, square, and triangle wave, respectively. If the channel is not switched on, the frames will be filled up with zeros. Additionally, after every iteration, the current angle is mapped from 0 to  $2\pi$  and stored in a buffer, allowing a smooth transition when the next thread is executed. The phase offset variable is set in degrees on the client-side and converted to radian on the native implementation.

```

if (ChannelONLeft) {
  if (WaveTypeLeft == 1) {
    BufferLeft[Frame] = SigNumber(
      sin(Controller->ThetaLeft - Controller->PhaseOffsetRadLeft))
      * AmplitudeLeft;
  } else if (WaveTypeLeft == 2) {
    BufferLeft[Frame] = ((2/M_PI) * asin(
      sin(Controller->ThetaLeft - Controller->PhaseOffsetRadLeft))
      * AmplitudeLeft);
  } else {
    BufferLeft[Frame] = sin(Controller->ThetaLeft - Controller->PhaseOffsetRadLeft)
      * AmplitudeLeft;
  }
  Controller->ThetaLeft += Controller->ThetaIncrementLeft;
  if (Controller->ThetaLeft > 2.0 * M_PI) {
    Controller->ThetaLeft -= 2.0 * M_PI;
  }
} else {
  BufferLeft[Frame] = 0;
}

```

Figure 14: The iOS implementation of the signal generator to fill the buffers

### 4.2.2 Oscilloscope (Foivos)

The oscilloscope required a constant stream of information from the native application into the Flutter project. This is a much different implementation to the signal generator, which only needed updates intermittently. This required a different kind of Flutter communication channel known as an `EventChannel`. This essentially allows the Flutter side of the project to create a listener that runs on an independent thread. The thread responds to a stream of information, rather than invoking a function. Doing this made it possible to handle the new information being returned from the microphone to display it without freezing the display.

Unfortunately, the Flutter `EventChannel` datatypes do not offer support for 16-bit signed lists to be transferred. Therefore, the 16-bit microphone stream is converted to an 8-bit with twice the number of elements. Table 2 demonstrates this operation, where  $X_N$

represents the 16-bit input buffer, and the two 8-bit lists  $Y_{N:1}$  &  $Y_{N:2}$  represent the transformation. The 8-bit list is then converted back into a 16-bit using the inverse operation once received on Flutter.

Table 2: Mapping of samples to the Flutter 8-bit event stream buffer

<i>Bit N</i>	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
$X_N$	$X_{16}$	$X_{15}$	$X_{14}$	$X_{13}$	$X_{12}$	$X_{11}$	$X_{10}$	$X_9$	$X_8$	$X_7$	$X_6$	$X_5$	$X_4$	$X_3$	$X_2$	$X_1$
$Y_{N:1}$									$X_8$	$X_7$	$X_6$	$X_5$	$X_4$	$X_3$	$X_2$	$X_1$
$Y_{N:2}$									$X_{16}$	$X_{15}$	$X_{14}$	$X_{13}$	$X_{12}$	$X_{11}$	$X_{10}$	$X_9$

#### 4.2.2.1 Android (Gavin)

The oscilloscope implementation on Android involved using and editing the Flutter package `mic_stream` [5]. This package uses native Android functions to provide a PCM stream at a specified sample rate, with a specified microphone source. The package uses `AudioRecord` [26], which records audio from the specified input hardware of the phone (`AudioSource.MIC` in this case) as seen in Figure 15. The sample rate is specified at the maximum value of 44.1 kHz as defined in the specification. The channel configuration is set to mono as the input signal received from the Music Mixer board is mono.

```
private int AUDIO_SOURCE = MediaRecorder.AudioSource.MIC;
private int SAMPLE_RATE = 44100;
private int CHANNEL_CONFIG = AudioFormat.CHANNEL_IN_MONO;
private int AUDIO_FORMAT = AudioFormat.ENCODING_PCM_16BIT;
private int BUFFER_SIZE = AudioRecord.getMinBufferSize(SAMPLE_RATE, CHANNEL_CONFIG,
    AUDIO_FORMAT);
```

Figure 15: Configuration of the `mic_stream` library for Android oscilloscope implementation

This recording implements PCM to record the audio, which is then sent in a stream to the Flutter side of the project. The minimum buffer size on Android phones is hardware dependent. As a result, the operation of `AudioRecord` is dictated by the sample rate, channel configuration and audio format. The minimum buffer size is calculated and used, as seen in Figure 15.

#### 4.2.2.2 iOS (Foivos)

Similar to the signal generator, the `AudioUnit` API was used for the oscilloscope to capture a PCM stream at a sample rate of 44.1kHz from the microphone input source. The recording unit is initialised as a single-channel 16-bit integer linear PCM for the mono output of the Music Mixer board. The `AudioUnit` for the oscilloscope is initialised in the same function as the signal generator. As the oscilloscope input is only a single channel, a 16-bit floating-point value was used. This is different from the audio output, which is 32-bit due to it being a dual channel.

Finally, once each sample buffer has been filled, an event is triggered that sends this information to Flutter. A listener is attached to the stream of the `EventChannel` to process the values for display on the GUI.

## 4.3 Home Screen (Leonardo)

### 4.3.1 Page Components (Leonardo)

Initially, for the design of the home screen, it was decided that the following components would be necessary: a top navigation bar with the title of the application (initially chosen as “EE Toolkit” but later renamed to “Electronics Everywhere”); button links to the respective kits; links to the websites for the UKESF [27], Electronics Everywhere [8] and University of Southampton outreach kits [9]. A bottom navigation bar was initially included to switch between the home, about and settings pages. However, this was later changed for a more compact home screen without a global settings page.

### 4.3.2 Initial Design and Implementation (Leonardo)

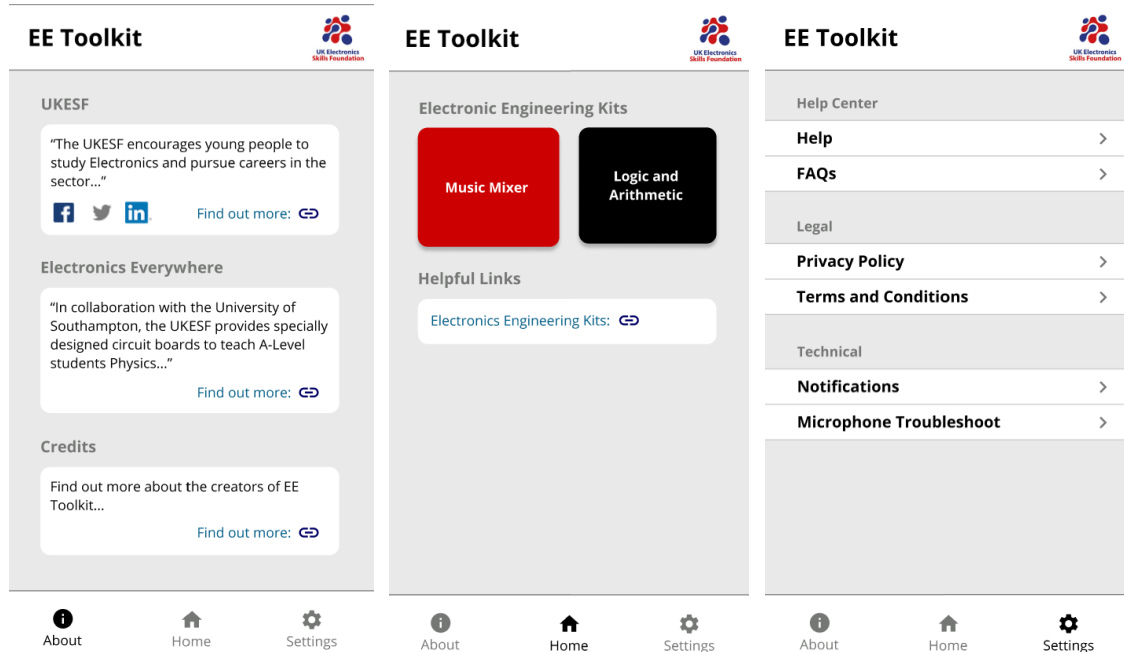


Figure 16: The original designs for the home screen showing (Left) the about page (Middle) the home page (Right) settings page

Figure 16 shows the initial design for the home screen in Figma. The original idea was to have a simple design with colours and styling similar to the UKESF website. It was initially decided to include an about page and a settings page with the home screen. This was decided from background research on other applications, as many applications have an about page and a global settings page (i.e. Facebook or Twitter). The initial design was implemented during the first two sprints, using the Scaffold widget, from the material components widgets, for the home screen. The AppBar widget was used for the top navigation bar and the BottomNavigationBar widget for the bottom navigation bar. For each page, custom widgets were made for similar components, for example, a custom container for a kit button with parameters for the background colour, text and linked page.

### 4.3.3 Design Changes and Considerations (Leonardo)

After discussing the initial design and its implementation, changes were made based on team discussion, questionnaire feedback, and practical implementation. One iteration included changing the whole colour scheme of the application and style to look more like

the UKESF styling found on the website. The decision was made as the initial design did not completely resemble the website and branding, that was essential for the client.

The change was implemented by adding a UKESF styled line widget using the CustomPaint widget. Images of the boards were added to the containers, and the about widgets linking to websites were branded using UKESF styling. The widgets used for theming are discussed further in 4.3.4.2.

After implementing the initial design, it was decided that there was no use for a global settings page for what was planned to be implemented. Therefore, the about page was merged into the main page to reduce space wastage, and the settings page was removed.

For concurrency throughout the application, the top navigation bar layout was changed so that the icon to the left of the title would either be the home icon or the UKESF logo and the right icon default to the settings cog.

#### 4.3.4 Final Design and Implementation (Leonardo)

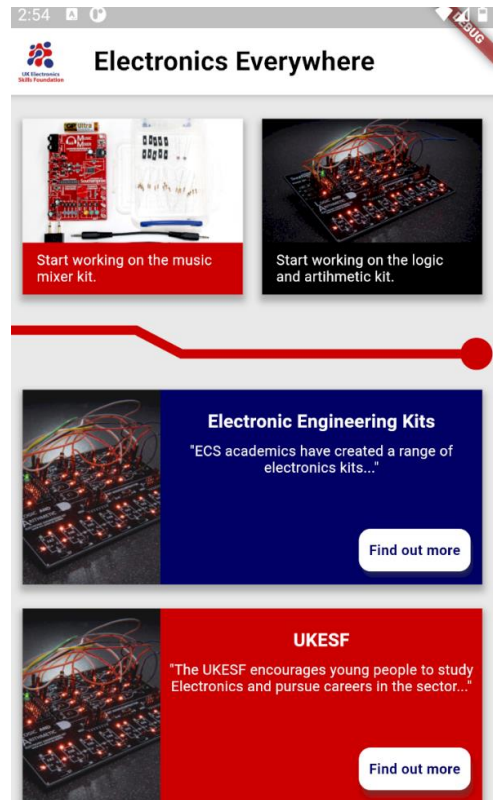


Figure 17: Final design of the home screen

The home screen is the landing screen for the user with the elements shown in Figure 17. The top of the screen has a navigation bar with the UKESF logo on the left and the application title. The main body of the page is split into two sections by a line widget with UKESF styling. The first section includes button links to the Music Mixer screen and Logic and Arithmetic screen; with an image of the respective kit and a brief explanation of the destination of the button. The second section includes three reference links to Electronics Everywhere, UKESF, and the Outreach Kits page from University of Southampton like the initial design. The home screen was implemented from the



StatelessWidget class, using the Scaffold widget with a custom AppBar for the top navigation bar and a ListView widget for the main body.

#### 4.3.4.1 Top Navigation Bar (Leonardo)

A custom AppBar widget was made for the top navigation bar throughout the application, with parameters for the title, theme name and optional settings parameters. Figure 18 shows how the top navigation bar is separated with the leading icon, title and action icon.

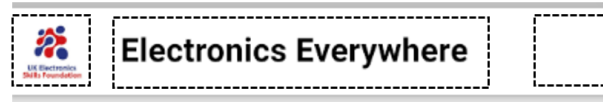


Figure 18: AppBar widget showing the separate parts

The background colour is set depending on the theme name, as either red, black or white. If the main theme is selected, the leading icon is set to the UKESF logo. If any other theme, then the leading icon is set to either the home icon or a back arrow depending on the optional parameters. If the page links to a settings page, then the cog icon is enabled.

The top navigation bar implements PreferredSizeWidget to set the height of the application bar to 56 pixels. An optional boolean parameter is used to determine whether the bar includes a settings page, and optional parameters for whether the page uses a back icon, with a parameter for the widget to go back to and the page index.

The Navigator API was used to move between pages – specifically the pushReplacement and push methods depending on whether the link was to the home page.

#### 4.3.4.2 UKESF Themed Widgets (Leonardo)

As shown in Figure 19, the kit containers were implemented using the Container widget wrapped in an InkWell widget. Inside the container is a Column widget for the image of the kit and a brief description. Each of the containers has the width set to half the width of the screen minus 20, to account for the padding. The height of the image container was set using the AspectRatio widget set to a ratio of 16:9.

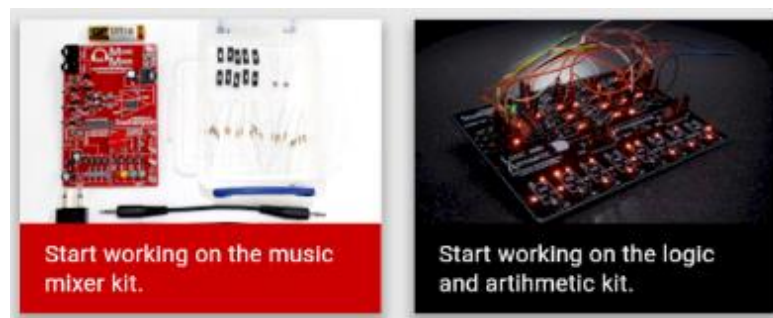


Figure 19: Container widgets found on the home screen for the kits

The about containers, which link to the Electronic Everywhere kits page; UKESF home page; Electronics Everywhere page, were styled using UKESF theming from the website, shown in Figure 20. The “Find out more” button is implemented with the Stack and Positioned widgets to show on the container. The width distribution of the image and text used the Expanded widget, with a flex of three for the image and flex of seven for the text portion. The package url\_launcher [2] was used with the functions canLaunch and launch inside an asynchronous function, with the await property to open the website in-browser.

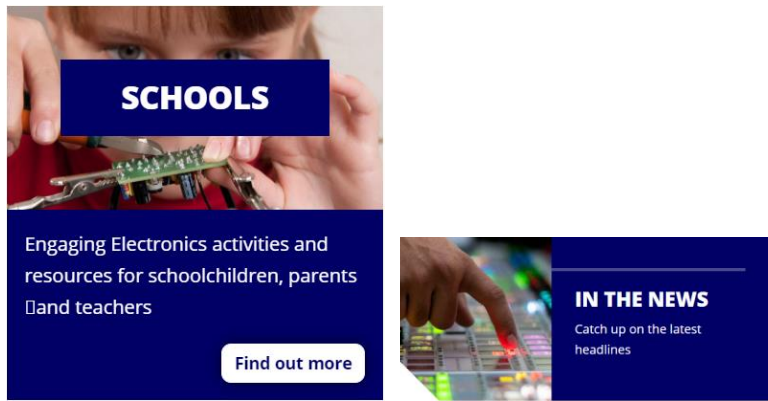


Figure 20: Examples of the styling from the UKESF website (sourced [27])

The coloured line UKESF styled widget was made using the CustomPainter widget with several different parameters – including the colour, size and positioning. There are three offset parameters to position the widget, including the left and right offset, and height offset. Optional parameters are used to produce a diagonal line (if that is desired). Two percentage parameters are used to determine the percentage of the width to start and end the deviation, and a parameter is used for the height, as shown in Figure 21. The line is drawn using the Path class and canvas to draw the line and the circle at the end of the line.



Figure 21: PCB trace widget taken from (Left) the UKESF website [27] (Right) the application

#### 4.3.4.3 Colour Scheme and Font (Leonardo)

The UKESF website uses a white background behind different coloured widgets, so it was decided to use a white navigation bar against a slightly grey background. The only significant colours chosen for the home page were red and black for the kit link button, to resemble the actual Electronics Everywhere kits. The primary colours used for text in the UKESF website are white, grey, and black depending on the background colour and the font used for the application was also chosen to be Open Sans as on the website.

## 4.4 About Page (Leonardo)

### 4.4.1 Page Components (Leonardo)

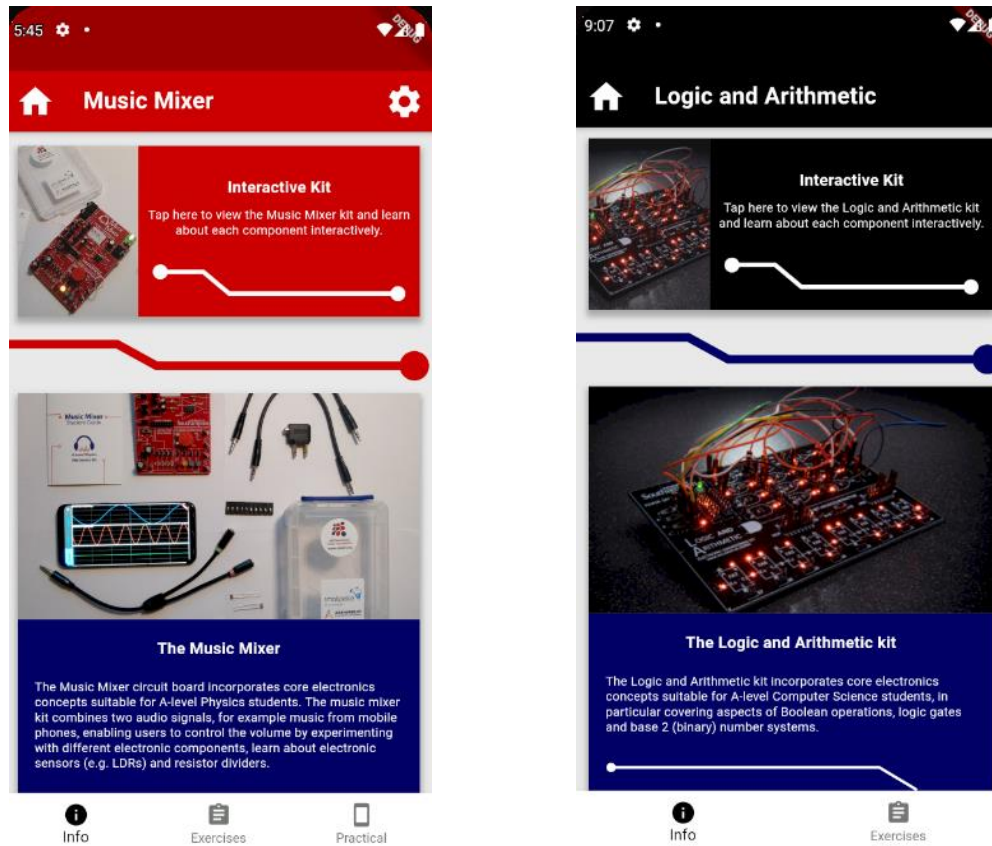


Figure 22: About page for (Left) the Music Mixer (Right) the Logic and Arithmetic

The about page includes a container widget, which links to the respective interactive image page for the kit, and a widget with an image and information about the kit. These two parts are separated with a line widget, as shown in Figure 22.

### 4.4.2 Design Decisions (Leonardo)

Originally from the initial design, an about page was not included. However, it was decided to include it to have information about the equipment and a link to the interactive image of the kit. The colour of the AppBar was chosen to resemble the colour of the respective kit, red for the Music Mixer section, and black for the Logic and Arithmetic section.

### 4.4.3 Implementation (Leonardo)

The about page includes a line UKESF custom widget separating two containers – one that takes the user to the interactive image page and another that contains information about the Music Mixer kit.

Both widgets are styled with UKESF theming. The top link widget is styled similar to the widget on the website shown in Figure 23. However, after feedback, the section was changed to have an angle to avoid being confused for a slider.



Figure 23: Button widget from (Left) the UKESF website [27] (Right) the application

The bottom widget includes information about the kit is styled like one of the widgets on the website, as shown in Figure 24. However, the style is altered to fit a phone screen better. It does not link to any other page.

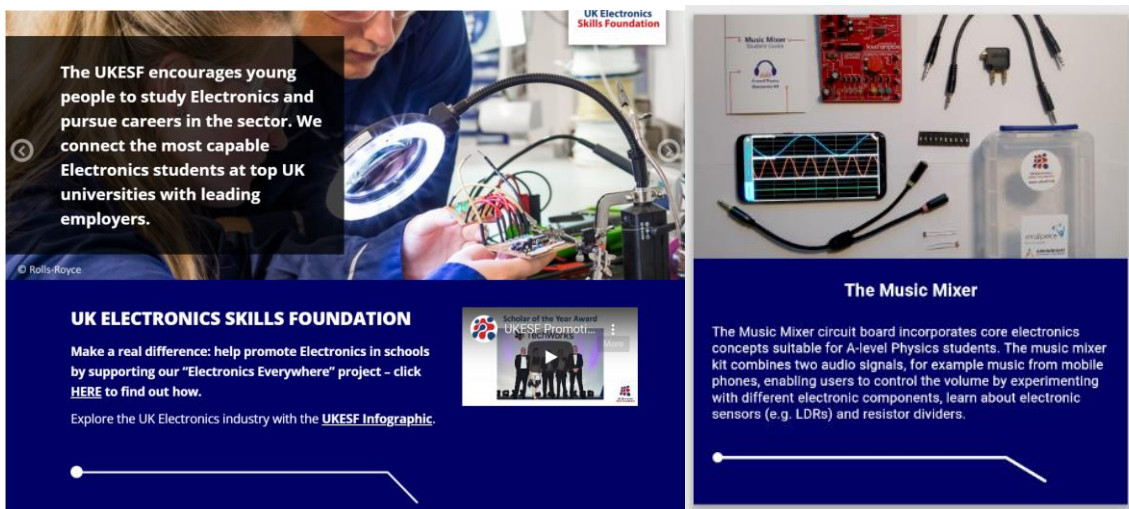


Figure 24: Information widget from (Left) the UKESF website [27] (Right) the application

#### 4.4.4 Music Mixer Screen (Leonardo)

After pressing the Music Mixer kit container on the home page, the top navigation bar styling changes to the Music Mixer theme. The default white is changed to the red used on the website to line with the colours on the physical kit and website. The logo is replaced with a home icon allowing the user to return to the home screen. There is also a settings icon to change the settings for the practical page. The bottom navigation bar switches between the about page, exercise page, and practical page.

The Music Mixer screen is implemented similar to the home screen, but with changed parameters for the AppBar widget, and includes a BottomNavigationBar widget to switch between tabs by updating the current index.

#### 4.4.5 Logic and Arithmetic Screen (Leonardo)

The about page for the Logic and Arithmetic screen is the same but with different parameters for the custom widgets such as colour, image and text content. The screen itself only has two tabs for the bottom navigation bar: The about page and exercises page. It also does not have a settings page, as shown in Figure 22.

## 4.5 Interactive Image Page <sup>(Adel)</sup>

### 4.5.1 Page Content <sup>(Adel)</sup>

The original idea for the interactive image page was to include a section where users can familiarise themselves with the available boards and discover the electronics and concepts at their core.

The initial design described the page initialising as an image of the selected board filling the screen, with a transparent button stacked above the image for each section of the board. If the user pressed this button, a tooltip should appear displaying simple information about the board. An additional information page for each of the sections/component of the two boards can be opened from this tooltip. The page should also allow the user to zoom and pan the image. These pages show the name and a brief description of the component and explain the electronics behind the scene. Figure 25 shows the initial design of the interactive image and the additional information page for the NAND Gate on the Logic and Arithmetic board.

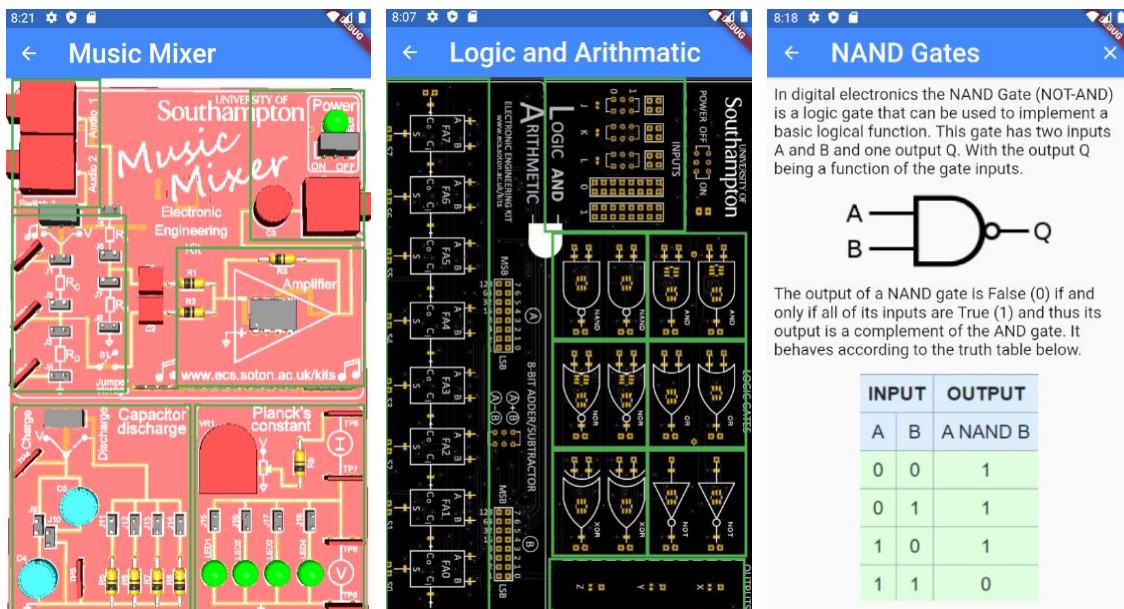


Figure 25: Initial design of the interactive pages for (Left) the Music Mixer (Middle) the Logic and Arithmetic (Right) the additional information

In the initial design of the interactive image page, a default application bar was used with a back button which would return the user to the previous page. The body of the page was a container that used the Stack widget to layer buttons on top of the image. The tooltips were not yet implemented, however, links to each section of the additional information page of the boards were implemented using a FlatButton widget was used with the colour set to transparent with a green border to indicate each section. A Positioned widget was used to correctly place the button, with the position and size of the buttons being hardcoded. The onPressed function of each button navigated to the relevant additional information page.

The additional information pages were created as a MaterialPageRoute class that would return a Scaffold of an AppBar and body. The body was comprised of Column widget with its children being the text and image widgets. The ScrollView widget was used to

enable the body to overflow the screen and enable scrolling feature. A `SizedBox` widget was used to create spacing between the texts and images.

### 4.5.2 Design Changes (Adel)

After testing the initial implementation on multiple devices, some issues with the text and image sizes were found. The sizes and positions being hardcoded meant on devices with different screen sizes, the size and position of the buttons were out of sync with the image of the board. This resulted in the button not overlaying its corresponding section. A similar issue was found in the additional information pages where the text and images would be too small on larger screens and too big on smaller ones. It was found that the canvas size of some of the image files used was too small, and Flutter was restricted in displaying the images. The size and position of elements in the application needed to be changed to be relative to the host device screen size to fix this. It was also decided that `SizedBox` widget used to create spacing should be removed and replaced with padding and margin added to text and image containers. These padding and margins should also be relative to the size of the screen.

One of the missing features was the ability to zoom into the image and pan around. This was important to allow seeing all the elements on the board more clearly. The green borders of the buttons did not align with the branding of the application, requiring a change. The implementation had to clearly show the sections that can be interacted with without distorting the image.

Another missing feature was an intermediary tooltip page between the interactive page and the additional information pages. The additional information should appear in both the tooltip and on its independent page. The tooltip would have several pages to contain all the information of the additional information page. The additional information page would be mostly unchanged from the initial development, apart from adding some padding and animations.

### 4.5.3 Final Design and Implementation (Adel)

To ensure the images are the correct size for the display, the size of the screen must be determined. This is done with the help of the `MediaQuery` function. When called, this function returns data about the current media, such as the screen size. One concern about using the `MediaQuery` function is that it provides the full-screen sizes, not accounting for any toolbars. To account for this, the `AppBar` and toolbar heights can be subtracted to get the total available space for the image.

Once the maximum size is determined, it is possible to use the same algorithms initially hardcoded. This was done by replacing the elements hardcoded height and width with a fraction of the height and width of the page. The text size and image size of the additional information pages were also set relative to the screen size. The `sizedBox` widgets were also removed and replaced with container widgets with padding that included the text and images. This means that the layout will stay consistent irrelevant of the device used.

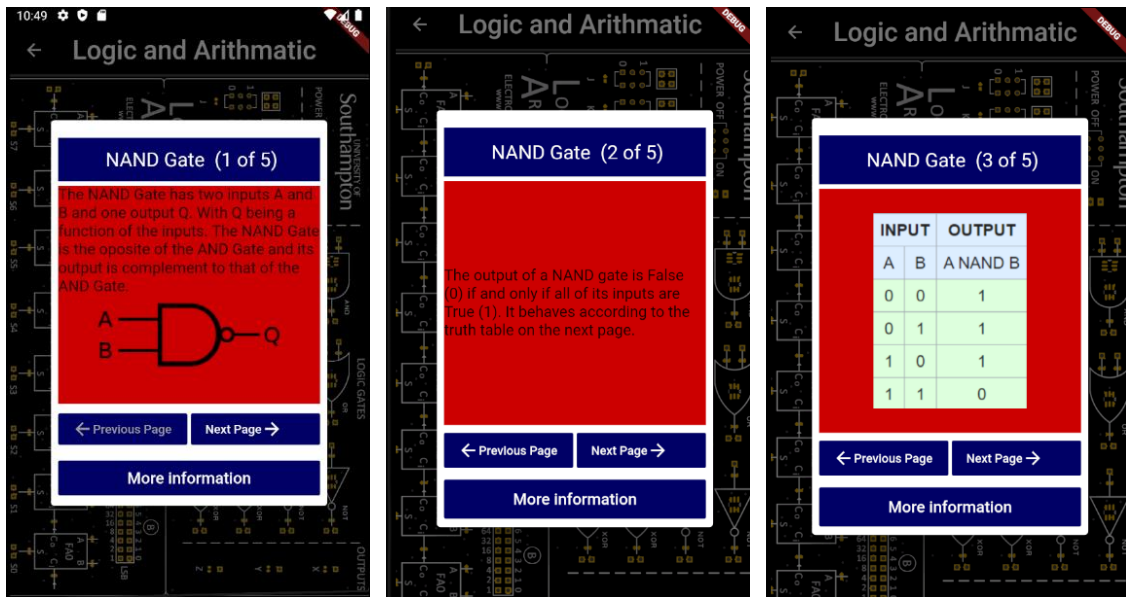


Figure 26: Final design of the tooltip for the interactive image page

For the tooltip design, multiple methods were considered, including `AnimatedBox`, `InkWell` and `AlertDialog` widgets. The `AlertDialog` widget method was chosen as the built-in functionalities created a more coherent and user-friendly interface. The final design of the tooltips can be seen in Figure 26.

All the content and the name of the tooltips were stored using a map. A custom tooltip class generates the tooltips to display. This class receives the name of the tooltip and returns an `AlertDialog` widget, containing a column widget with three containers as children. The first container displays the name of the tooltip and the number of available tooltips. The second is a container which fetches its content from the map, and the third container consists of the previous page and next page buttons which when pressed which cycle through the tooltips by setting the state of the tooltip.

The `AnimatedSwitcher` widget was used to smooth the transition between the tooltips. The `AlertDialog` action was set using the more information button which would route to the additional information pages. To incorporate the UKESF theme, the red, blue and white colours of the UKESF website was used for the tooltips.

In the interactive image page, the zoom and pan features were added using the `interactiveViewer` widget with a minimum scale of one and a maximum scale of five. To indicate the sections of the board that can be interacted with, a custom animation was used. The animation uses a timer that cycles through each section of the board image and highlights it by changing the opacity of the button from zero to one and back to zero before moving to the next section. This highlights each interactable section of the board. Global variables are used for the duration of the animation and the time between animation completion and restart. Figure 27 shows all the Logic and Arithmetic page sections and how each section will look when they are highlighted. Each adjacent section is highlighted in alternating UKESF colours (red and blue) to align with the UKESF theme.

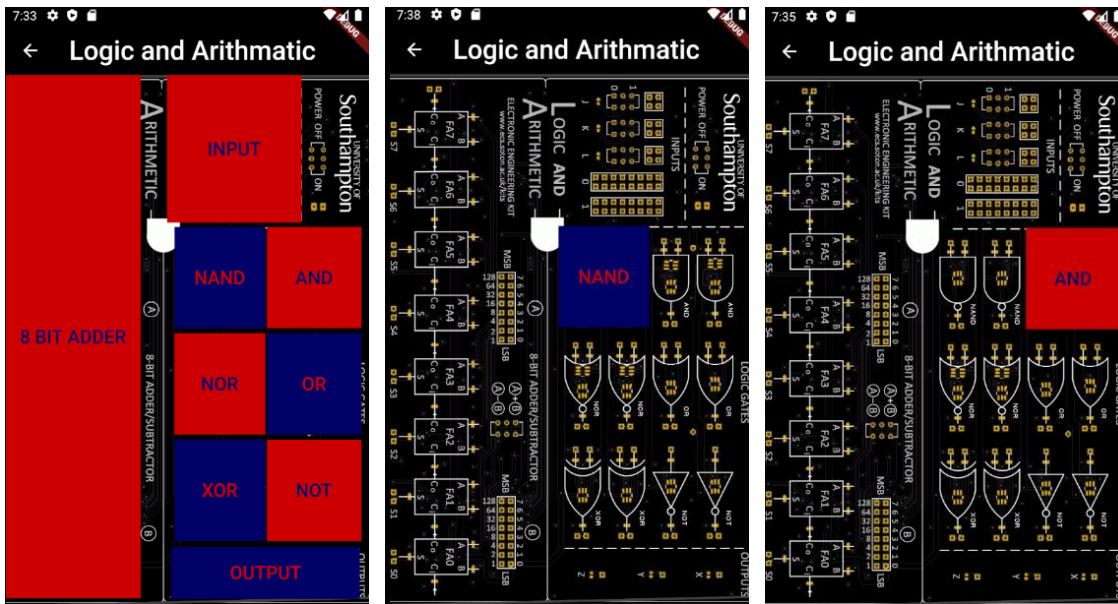


Figure 27: Tooltip animation showing the passage of time from left-to-right

Another addition was replacing the circuit diagram and truth table of the gates in the Logic and Arithmetic additional information pages with a custom rotatingImage widget. This widget takes several images and cycles through them. The images used were edited versions of the original images so that when cycled through, they show the gates for different inputs. The rotatingImage widget has the advantage of using less memory than GIFs or video as it only uses five images. Figure 28 shows how the rotating image widget cycles through for the NAND Gate additional information page. The circuit diagram also cycles through all different inputs, and the truth table starts empty and fill up and cycles through. The speed at which the widget switches between the images can be modified through a global variable.


In the final design of the interactive images page, the starting point is the image spanning the page where the user can zoom in and pan around the board. The custom tooltip animation will cycle through all the board sections indicating that what is interactable. Once the user interacts with one of the sections, the corresponding tooltip will give the user useful information about the section and its use. The next page and previous page can be used to navigate through the tooltips. The more information button will take the user to the additional information page to find out more about the physics and electronics operation.



8:44

### NAND Gates

In digital electronics the NAND Gate (NOT-AND) is a logic gate that can be used to implement a basic logical function. This gate has two inputs A and B and one output Q. With the output Q being a function of the gate inputs.




The output of a NAND gate is False (0) if and only if all of its inputs are True (1) and thus its output is a complement of the AND gate. It behaves according to the truth table below.

INPUT		OUTPUT
A	B	A NAND B
0	0	1

9:51

### NAND Gates

In digital electronics the NAND Gate (NOT-AND) is a logic gate that can be used to implement a basic logical function. This gate has two inputs A and B and one output Q. With the output Q being a function of the gate inputs.




The output of a NAND gate is False (0) if and only if all of its inputs are True (1) and thus its output is a complement of the AND gate. It behaves according to the truth table below.

INPUT		OUTPUT
A	B	A NAND B
0	0	1
0	1	1

9:53

### NAND Gates

In digital electronics the NAND Gate (NOT-AND) is a logic gate that can be used to implement a basic logical function. This gate has two inputs A and B and one output Q. With the output Q being a function of the gate inputs.




The output of a NAND gate is False (0) if and only if all of its inputs are True (1) and thus its output is a complement of the AND gate. It behaves according to the truth table below.

INPUT		OUTPUT
A	B	A NAND B
0	0	1
0	1	1
1	0	1

9:57

### NAND Gates

In digital electronics the NAND Gate (NOT-AND) is a logic gate that can be used to implement a basic logical function. This gate has two inputs A and B and one output Q. With the output Q being a function of the gate inputs.



The output of a NAND gate is False (0) if and only if all of its inputs are True (1) and thus its output is a complement of the AND gate. It behaves according to the truth table below.

INPUT		OUTPUT
A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0

Figure 28: RotatingImage widget for NAND Gate additional information page (cycle goes top-left, top-right, bottom-left, bottom-right)

## 4.6 Exercises Pages (Leonardo)

### 4.6.1 Page Components (Leonardo)

At the start of the project, it was decided to include a page where the user could view the existing lab notes from within the application. This page should support tabs to select which lab to follow; a link to the existing lab notes; a separate page for each section in the notes to follow along step by step.

### 4.6.2 Initial Design and Implementation (Leonardo)

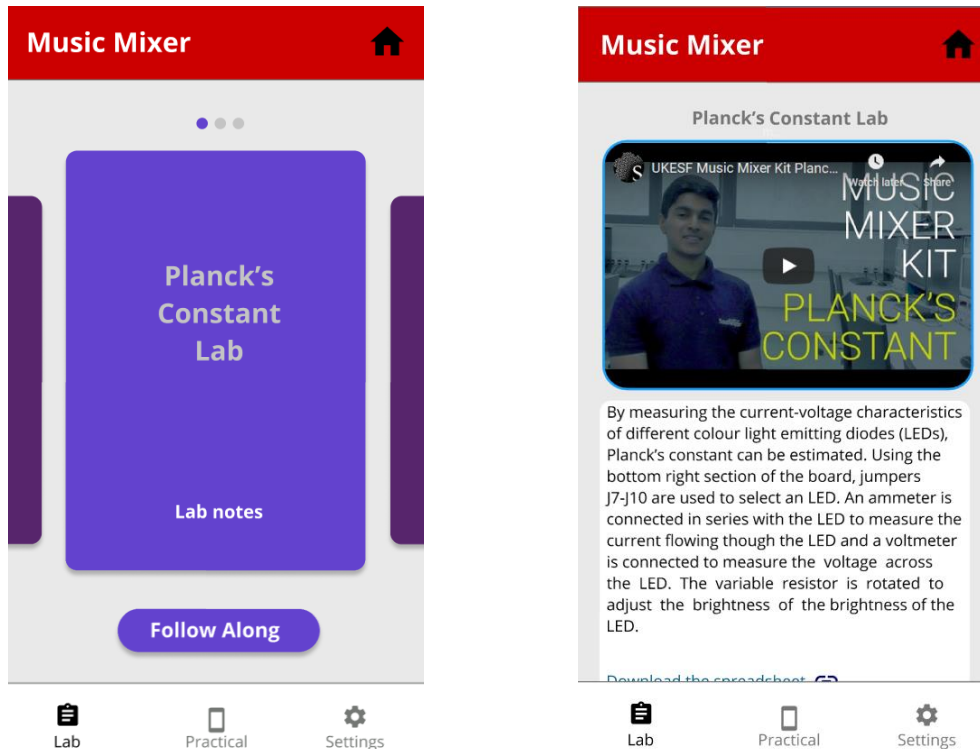


Figure 29: Initial Design for the exercise page

Figure 29 shows the initial design for the exercise page for the Music Mixer screen, where the user first chooses which lab to follow along. For example, after selecting the Planck's Constant Lab, the user is redirected to a page showing information about the lab, with an embedded video about using the Planck's Constant part of the kit. In that page would be a button link to another page with steps for each section.

The initial design was implemented using the PageView widget builder with a custom container for the lab information. A template lab container was created so that a list of labs would be used to reduce code repetition. The animation of the PageView slider was made using a controller with a listener that updates a list with the extra padding for each lab container so that when swiping through the labs, the selected lab container would pop out. The list is updated so that the lab on the current page has no extra margin but those not on the current page have an extra margin.

### 4.6.3 Design Changes and Considerations (Leonardo)

Numerous changes were made to the initial design of the pages based on practical implementation and team discussions. The theming of the page changed to fit more in line with the UKESF branding to the entire application. For example, the purple was replaced

with blue and red, and the rounded corners were replaced with rectangular ones. For each of the four lab sections, an image of the kit being used with the lab was added to the containers. The design of the second information page was also changed in this respect.

The primary button was swapped from “follow along” to the lab notes source to focus on the existing lab notes. The entire container was made pressable, using the InkWell widget, instead of just the “follow along” button to make it easier for the user to choose a lab section to follow.

The tab animation was smoothed out so that the container transitions would not be instantaneous. A steps section was added after choosing a lab for the user to follow along easily. This design choice was made to make following along to the lab notes more engaging.

After deciding to include exercise pages for the Logic and Arithmetic section, a new design had to be made to resemble the training handbook, as the styling was different to the template widgets made for the Music Mixer.

#### 4.6.4 Final Design and Implementation (Leonardo)

The final design for the exercise pages is shown in Figure 30, with all the changes discussed in 4.6.3.

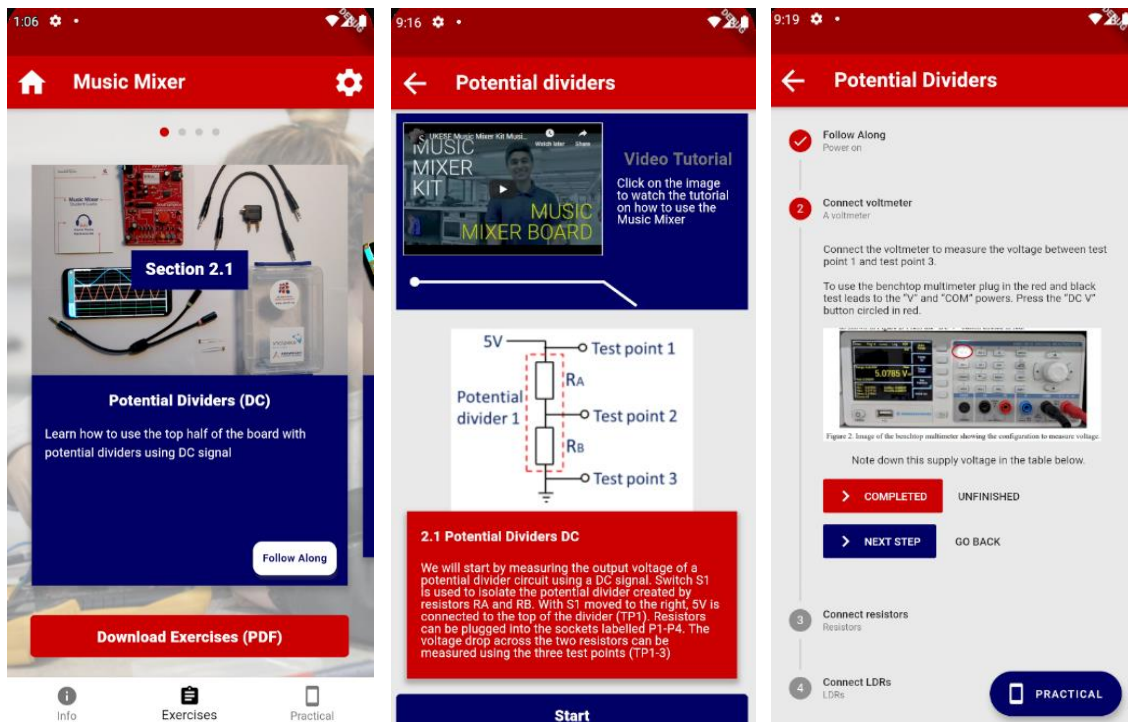


Figure 30: Final design for the exercise pages

##### 4.6.4.1 Lab Selection Page (Leonardo)

The first page for the lab selection includes the page view, where the user may swipe between containers for different sections from the lab notes training handbook [28]. Figure 31 shows the smooth transitions between the sections.

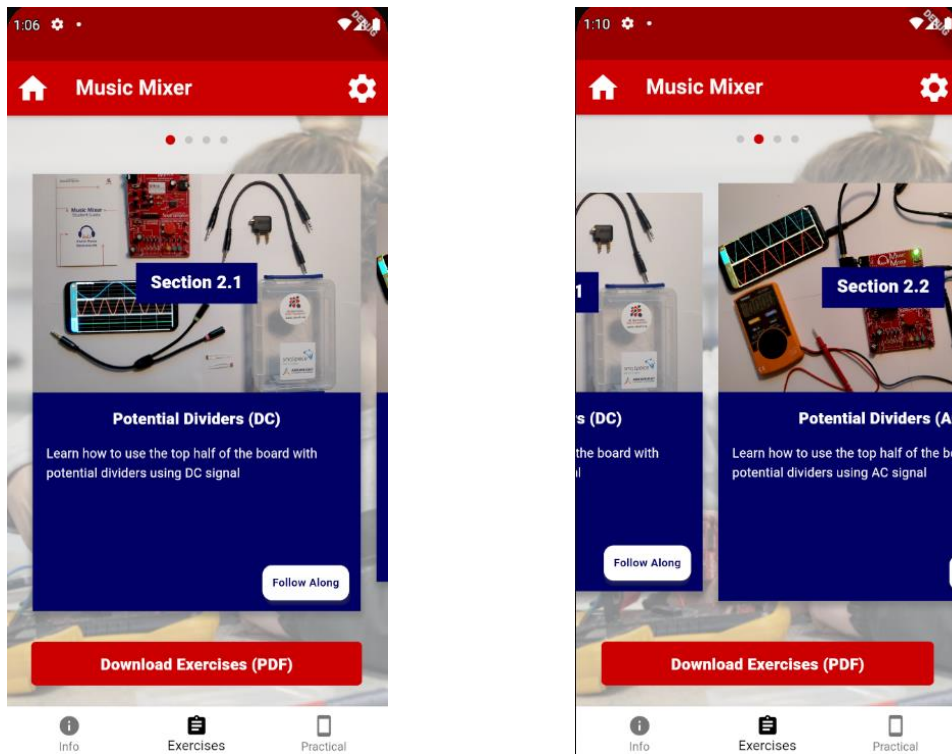


Figure 31: The exercises containers (Left) static (Right) in a transition

```

for (int i = 0; i < labList.length; i++)
    _margins[i] = (i == _currentLab ? 0 : margin);
_controller =
    PageController(initialPage: 0, viewportFraction: viewportFraction)
    ..addListener(() {
        setState(() {
            _currentLab = _controller.page.round();
            for (int i = 0; i < labList.length; i++) {
                if (i <= _controller.page + 1 && i >= _controller.page - 1) {
                    if (i >= _controller.page)
                        _margins[i] = margin - ((_controller.page - (i - 1)) * margin);
                    else if (i < _controller.page)
                        _margins[i] = ((_controller.page - i) * margin);
                } else {
                    _margins[i] = margin;
                }
            }
        });
    });
});

```

Figure 32: Code for updating the padding for each of the lab containers

The lab container was implemented similarly to the about widget in style but with extra features. It takes the parameters of index, margin and list of the labs. Each lab includes a section name, title, description, image and next screen. Each row has a flex number of five, so the height of the image and body are the same. The top and bottom padding are set by the margin variable, which updates depending on the current page. The code used to calculate the margin for each page is shown in Figure 32.

Above the section containers are dots to show the positioning of the current section index. It was implemented with a custom widget for each dot, returning an `AnimatedContainer` with a duration of 200 milliseconds. The dots have parameters for the colour and a boolean to determine if the dot is active. If the dot is active, then the height is increased, and the colour changed.

The button at the bottom of the page links to the source training handbook where the pdf may be downloaded. Figure 33 shows the downloading of the lab notes after pressing the button. The button was implemented using a custom widget returning a `FlatButton` wrapping in a `Padding` widget, with parameters for the text, colour and source notes URL. It uses the `url_launcher` package to open the URL in the browser.

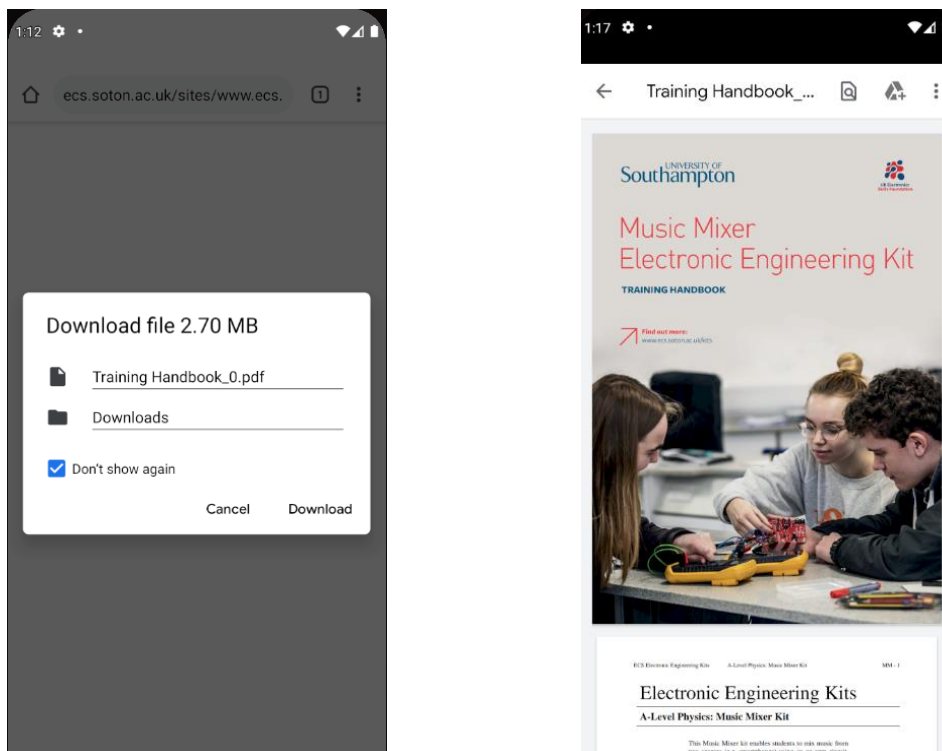


Figure 33: Training handbook download after pressing the button on the Music Mixer lab page

#### 4.6.4.2 Lab Information Page (Leonardo)

A page for each of the four lab sections was implemented as a `StatelessWidget`, shown in Figure 34. The first widget on each page has a link to the website with the tutorial video shown in the image and brief explanation. Under the video link widget is an image relating to the lab followed by brief start instructions from the training handbook. There is a button to start the lab at the bottom, which redirects the user to the step-by-step instructions shown in Figure 35.



Figure 34: The four introduction pages for each section of the Music Mixer lab

This page was implemented using a custom StatelessWidget that takes parameters for the title, header, video link text, URL, image, description, description image, page index and page link. The body of the page is built with a ListView widget with elements for the video link, text and button.

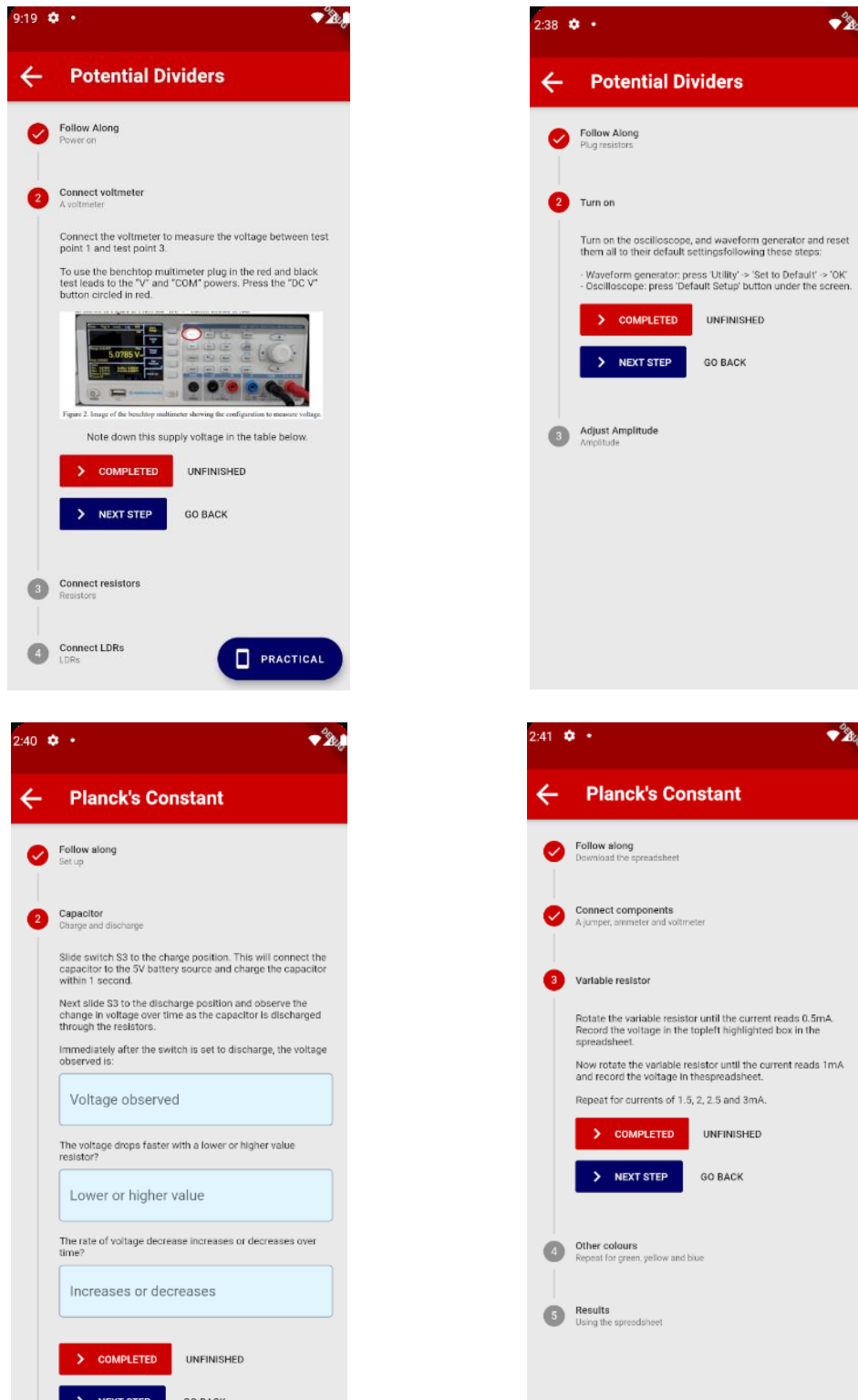


Figure 35: The four step-by-step pages for each section of the Music Mixer lab

Each of the pages has a link to a different StatefulWidget, as shown in Figure 35. The parameters of this widget are the title, step-index, number of steps, and a list of the completed steps. Most importantly is a list of Step widgets which form the actual body of the page.

A step is updated to active if its index is greater than or equal to the index of the viewed step. Its state is set by a list which is updated when the completed button is pressed. If the next step button is pressed, then the index is increased. The opposite is done for the unfinished and “go back” buttons.

#### 4.6.5 Logic and Arithmetic Screen (Leonardo)

The exercise page for the Logic and Arithmetic screen was done after that of the Music Mixer screen, so the main lab selection page was implemented the using the same widgets used for the Music Mixer screen with different arguments as shown in Figure 36.



Figure 36: Exercises page for the Logic and Arithmetic screen

The logic problems [29] were used to model the design and content of the pages, as shown in Figure 37. The colours used are the same as in the source pdf, and the structure is the same. The main difference is the solutions at the end of the page that may be unhidden when the user wants to see the solutions. Every solution for every problem has three steps, like in the source pdf.



**Problem 2: Private Collector's Trap**

A private collector has received a valuable gemstone which they wish to put on display. Due to its value, the collector has proposed a trap to prevent thieves from stealing the gemstone or escaping after attempting to steal it.

The gemstone rests on top of a pressure plate on a pedestal, surrounded by a glass case. If the glass is broken and the gemstone's weight is removed from the pressure plate, the trap is set off. A steel barrier is then lowered, blocking the only entrance to the room and trapping any thieves.

Design the logic circuit required for this trap, modelling the glass case, pressure plate and steel barrier as follows:

- Glass case (Input J): 0-Glass not broken, 1-Glass broken
- Pressure plate (Input K): 0-Weight removed (gemstone missing), 1-Weight applied (gemstone not missing)
- Steel barrier (Output X): 0-Barrier raised (entrance open), 1-Barrier lowered (entrance blocked)

**Solutions**

- 1 Create a truth table:
 

J	K	X
0	0	0
0	1	0
1	0	1
1	1	0
- 2 Derive the logic equation:
- 3 Create a logic circuit:

Figure 37: An example of the problem exercises for Logic and Arithmetic (Problem 2)

Like the Music Mixer labs, there is a list of custom lab page StatefulWidget that take parameters to make the page with solutions. As parameters, it takes the title, the header (in the orange container), the image, the description, page index, the question, the solution steps and the table.

**Private Collector's Trap**

Steel barrier (Output X): 0-Barrier raised (entrance open), 1-Barrier lowered (entrance blocked)

**Solutions**

- 1 Create a truth table:
- 2 Derive the logic equation:
 
$$X = J \cdot \bar{K}$$

This equation is reached by reading all the combinations of inputs that result in  $X = 1$  from the truth table. There is only one combination in this case, so this combination is the solution.
- 3 Create a logic circuit:

Figure 38: Image popup on the problem exercises for the Logic and Arithmetic

The body of the page is implemented with a ListView widget for the different sections. The Table widget was used to recreate the table used in the pdf. As the information in the solutions might be hard to read in smaller devices, a popup was added using the Dialog widget for the images in the solutions. This is shown in Figure 38.

## 4.7 Settings Page (Leonardo)

### 4.7.1 Page Components and Design Considerations (Leonardo)

At the start of the project, it was decided that the main components in the settings page would be general settings for the entire application. However, this design changed as only two settings were implemented that could be changed. The first setting is the practical page graph theme, which can either be light or dark. As most physics students might not be used to the oscilloscope axis style and would be more accustomed to white background styling, the second setting would toggle the axis style to resemble an oscilloscope or a common axis. The switched-on colour was also changed to be the same blue used on the website and throughout the application.

### 4.7.2 Final Design and Implementation (Leonardo)

The final design of the page is shown in Figure 39. The design did not change as much for the body of the page and uses a Switch widget to toggle the setting in the practical page. The setting containers themselves were implemented with the ListTile widget with the Switch widget as the tile trailing widget. When the setting is changed, the values are updated for both settings and the widget SharedPreferences is used to update the practical page preferences. The discussion surrounding the addition of these graph configurations and the graphical changes are shown in 4.8.4.2.

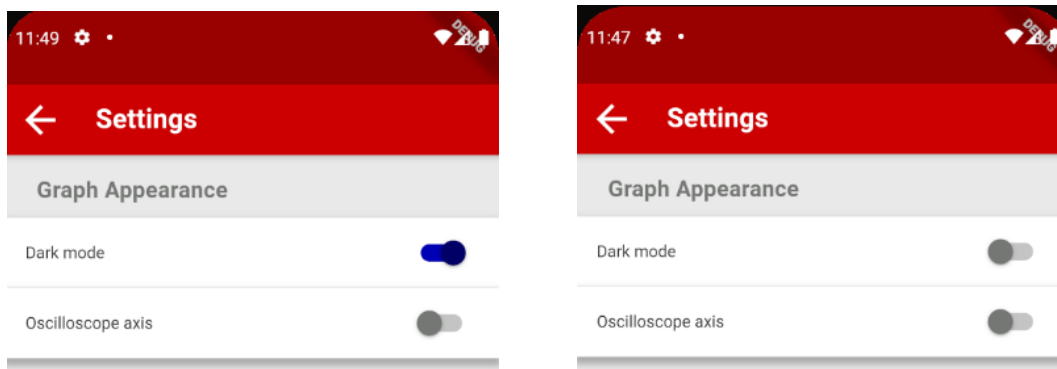


Figure 39: Final design of the settings page (Left) with dark mode enabled (Right) without dark mode enabled

## 4.8 Practical Page (Gavin)

The practical page for the Music Mixer kit contains the tools and interface needed to interact with the Music Mixer board for experiments. This section is where students would spend the most time, as they would be controlling the tools and viewing the signals all from this page.

### 4.8.1 Design Considerations (Gavin)

As the user will be interacting with the practical page for any interactive experiments, some considerations were made before development. Firstly, the oscilloscope and signal generator should be able to run independently. There should be one view to see the signal

generated output, and another for the oscilloscope inputs. A third option to combine both the oscilloscope and signal generator so that a user can control the outputs and see the inputs simultaneously was named the “combined” mode. This “combined” mode allows users to run experiments using a single device while still showing the required information.

The practical page contains the signal generator, oscilloscope and combined modes and so a method for switching between them was required. One idea was the ability to slide left or right to change between tools, but this was dismissed as it might not have been intuitive to use or that it might not have been clear to users how to change between modes. Another idea was to display a sidebar that could be used to select each tool, that would be permanently visible and therefore indicate the selected mode. This was deemed the best solution for switching between modes and is further discussed in 4.8.3.

Both the signal generator and oscilloscope have parameters (or settings) that need to be displayed and changeable. In other applications, such as the Keuwlsoft signal generator application [21] shown in Figure 8, displays the settings permanently on the screen. One proposal was to have a parameter menu for each tool that is collapsible or expandable to maximise space for other parts of the page like the graph. This implementation is discussed in 4.8.3.

The parameters included in this application were carefully selected to provide a balance between functionality and ease of use. The fundamental features were required for the tools to aid in experiments, but some other features like Fast Fourier Transform (FFT) were outside of the scope of this application. The features to be included were discussed when producing the initial specification in 3.

#### 4.8.2 Page Layout and Overview Design (Gavin)

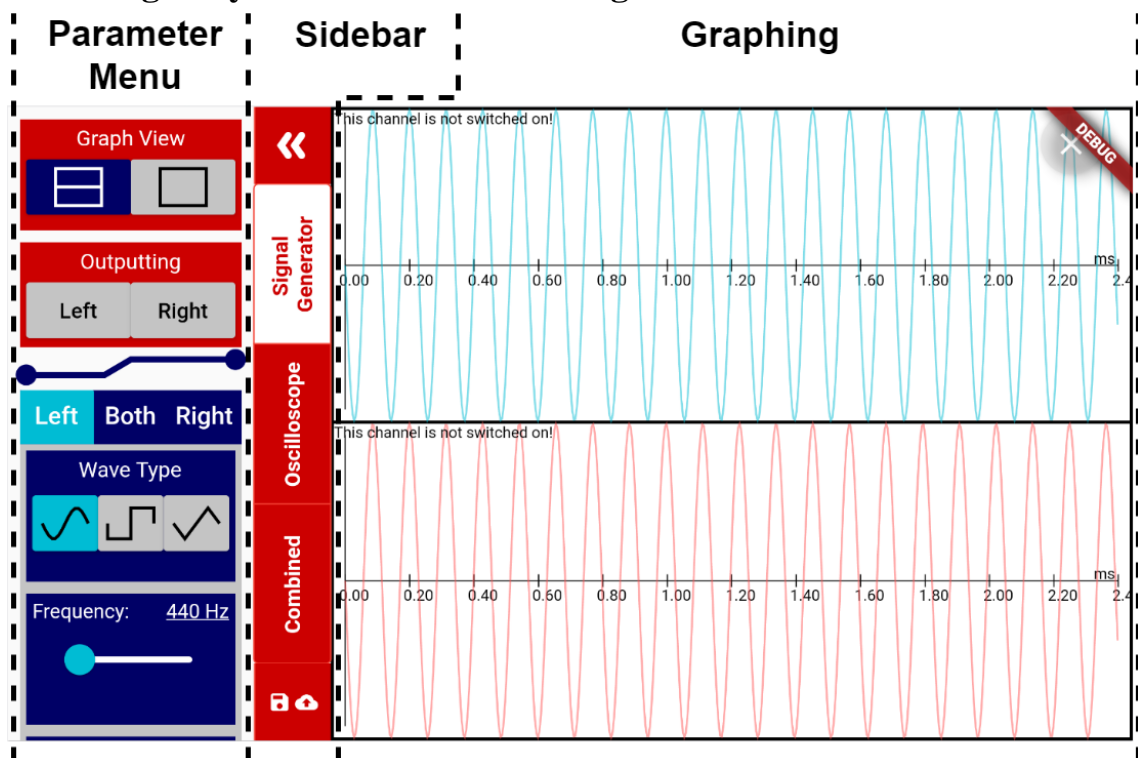


Figure 40: practical page with annotations of the three main sections

With the considerations in 4.8.1, the resulting design contains three main sections: the parameter menu, the sidebar, and the graphing. An annotated image can be seen in Figure 40.

As the priority of this page is running experiments, the graphing section should take up the majority of the screen. This allows the signals to be displayed as large as possible to give users the best view for understanding the concepts explored in the experiments. The parameter menu has been positioned such that it can be used while the graph is also visible. This menu is also scrollable to ensure the menu does not take up any unnecessary space. The mode currently selected and the other modes available are displayed in the sidebar to make it clear to the user what is available.

### 4.8.3 Sidebar and Parameter Menu (Gavin)

There is a corresponding parameter menu that is used to control the operation of the mode. The sidebar is a widget within the practical page that displays the mode selected and the button to access the save/load slots. The parameter menu can be collapsed or expanded by clicking the arrow section at the top of the sidebar. The sidebar slides in and out with the parameter menu, so the sidebar is always visible. When the parameters menu is collapsed, the sidebar is still visible. The parameter menu being expanded and contracted can be seen in Figure 41. The purpose of collapsing the parameter menu is to maximise the size of the graph section to give the user more detail to view. There is also an animation when collapsing/expanding the menu. The menu translates horizontally, while the arrow to collapse/expand the sidebar rotates 180° to indicate which way the menu can move.

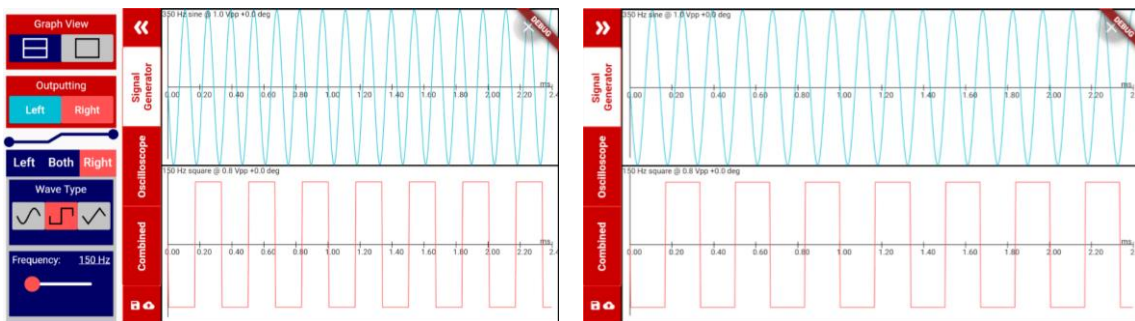


Figure 41: Parameter menu and sidebar on the practical page (Left) expanded (Right) collapsed

Implementing a collapsible parameter menu was done so the user can observe the effects that the parameters have on the signals displayed in the graph section while they are being changed. To see the changes made concurrently gives the user the best ability to understand and visualise the effect of changing certain aspects of the wave. When the user has finished changing the parameters, they can collapse the parameters menu to expand the space taken up by the graphs. This allows the user to observe the signals in the maximum size possible. When analysing other applications that contain similar tools, the controls took up the vast majority of the display and were permanent. However, in this application, it is essential to visualise the links between the parameters and the outputs. The Electronics Everywhere Application aims to aid learning and inspire students into the industry, and the typical user would not be familiar with these tools and their behaviours, so a more straightforward interface is required.

### 4.8.3.1 Signal Generator Parameter Menu (Gavin)

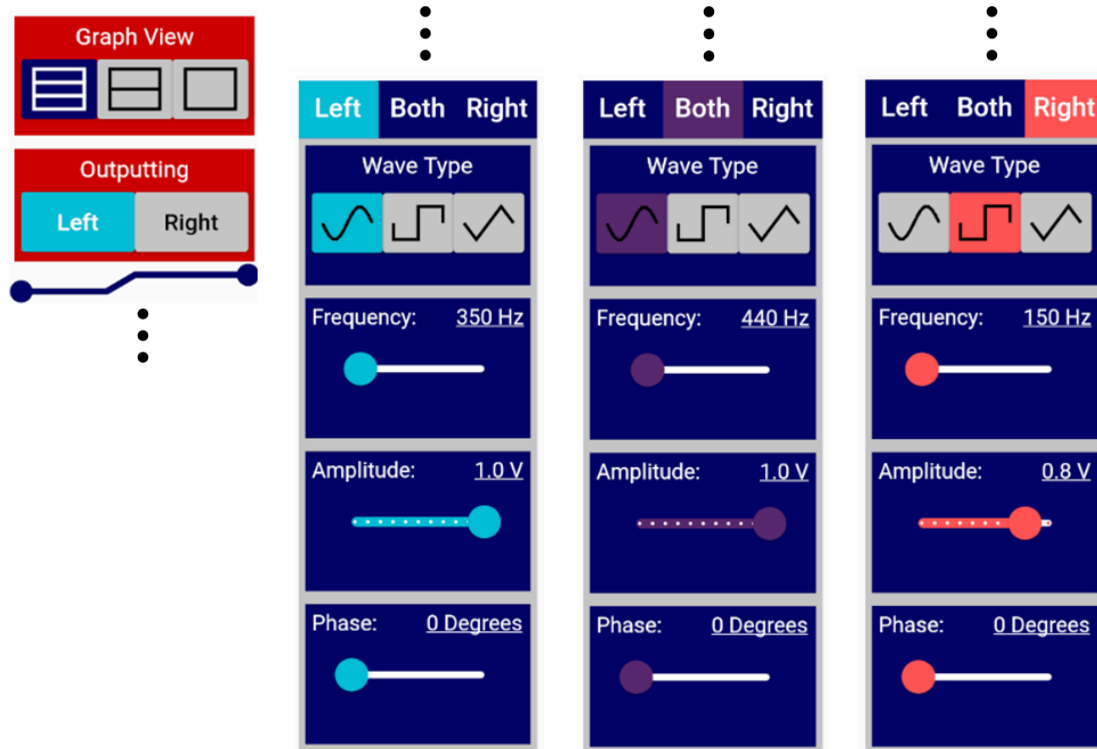


Figure 42: Signal generator parameter menu showing (Left) top section; bottom section with (Middle-Left) left selected (Middle-Right) both selected (Right) right selected

A consistent accent colour is displayed throughout the parameter menu corresponding to the signal affected by the buttons. For example, the left channel is displayed as light blue, accented for all the sections that control this signal, as shown by the sliders in Figure 42.

The “Graph View” box configures the graphing layout and will be discussed in 4.8.4. The “Outputting” box, shown in Figure 43, controls which generated signals are on or off (and being output to the auxiliary cable). The text changes from black to white and the highlight from grey to the trace colour to indicate a signal being on.



Figure 43: Outputting box with (Left) both signals turned on (Middle) left signal turned on (Right) none turned on

The application can generate two independent signals that correspond to the left and right channels. The third mode, “Both”, controls both the left and right signals simultaneously. Four parameters change the behaviour of each wave, the wave type, frequency, amplitude, and phase. To edit the parameters of a specific channel, for example, the left signal, there is a tabbed selection bar, as shown in Figure 44. When a tab is selected, the four parameters can be adjusted to change the property of the signal shown on the graph (and being output to the auxiliary cable). The parameter menu where each of the three signals is selected in the tabbed selection bar is displayed in Figure 42.



Figure 44: Wave parameter selection tabs while the “Left” channel is selected

The tabbed selection bar was chosen as this seemed the most intuitive method of changing between the selected signals. There is an animation when switching between tabs that slide between the tabs. This animation helps make it clear to the user that the parameters affect different signals. The accent colours in the parameter boxes match the tab colour to indicate further which signal was being edited.

The signal generator supports three wave types: sine, square and triangle. The “Wave Type” box controls these. There are three other boxes shown in Figure 45 that users can change the values with a slider or a text value entry.

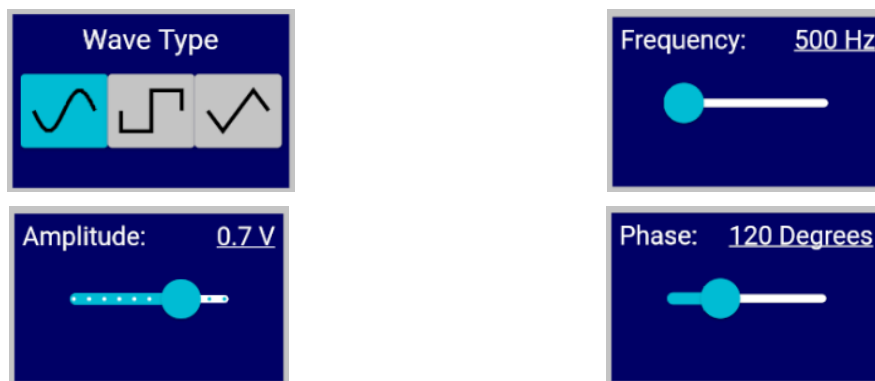


Figure 45: The user interface for editing (Top Left) Wave Type (Top Right) Frequency (Bottom Left) Amplitude (Bottom Right) Phase

The underlined values seen on the parameter boxes can be selected to open a text entry pop-up. An example of a text entry pop-up can be seen in Figure 46, which occurs after selecting the underlined part of the frequency box when the left signal was being edited.

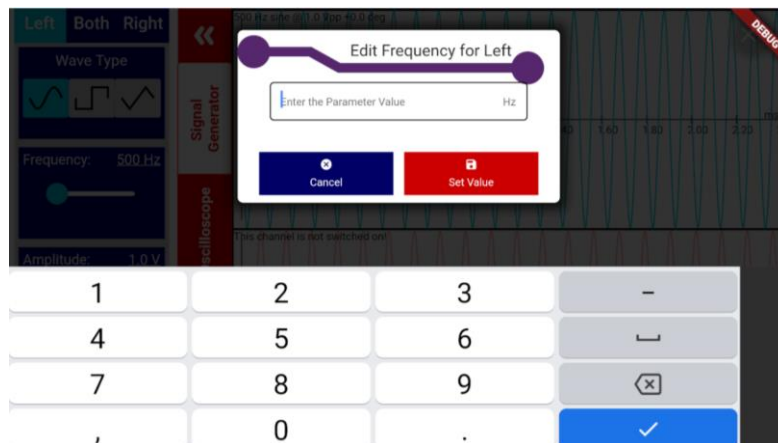


Figure 46: Text editing of the frequency for the left channel

The text entry allows for precise values to be inputted, giving the user fine control over the parameter. The values input must conform to unique format and value constraints for each variable type. The keyboard to enter the manual values is set to a numerical layout, seen in Figure 46. The values are checked to be within the supported range upon every keypress. For example, if the user typed 7000 for frequency, a message will remind the

user of the range allowed, as shown in Figure 47. Before any value is set, it is compared against the range to ensure only valid values are set. The units of the parameter are displayed at the right-hand side of the box.

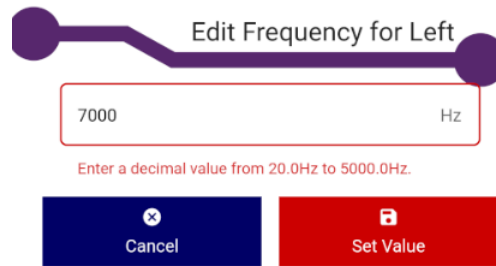


Figure 47: Attempting to input an out-of-range value into the frequency text entry

#### 4.8.3.2 Oscilloscope Parameter Menu (Gavin)

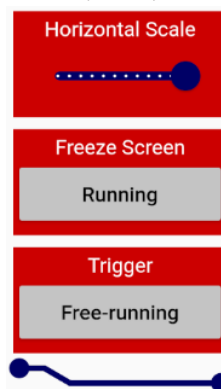


Figure 48: Oscilloscope parameter menu

The oscilloscope parameter menu can be seen in Figure 48. The horizontal scale controls the time duration shown on the graph at any given time. The slider has 11 discrete steps that correspond to the values in Table 3. The range of these durations has been chosen for specific reasons. The three largest time scales (12, 6 and 2.4 seconds) are used to show transient trends of more general features, since using the slowest 20 Hz signal generation would appear too many times (240, 120 and 48 times respectively). Beyond this, the scales are used to allow for good visibility of the signals at different scales. The rest of the scale is chosen as a time period of 1.2 seconds at 20 Hz and a 60 $\mu$ s time period at 5000 Hz will show 24 periods each.

Table 3: Horizontal scale steps and corresponding time duration values for graph

Step	1	2	3	4	5	6	7	8	9	10	11
Scale	1x	2x	5x	10x	20x	50x	100x	200x	500x	1000x	2000x
Time	12s	6s	2.4s	1.2s	600ms	240ms	120ms	60ms	2.4ms	120 $\mu$ s	60 $\mu$ s

A useful feature for observing an incoming signal is the ability to freeze the screen. This is controlled by toggling the button in the “Freeze Screen” box. The button displays “Running” when the incoming signal is displayed and displays “Frozen” then the signal has been frozen. The colour also changes to indicate if the mode is on or not.

Another feature discussed in the specification was to include triggering for the incoming wave. A periodic signal can be triggered by toggling the button in the “Triggering” box. The button displays “Free-running” when it is inactive and “Triggered” when it is. This functionality will be further discussed in 4.8.4.3.

### 4.8.3.3 Combined Parameter Menu (Gavin)

As shown in Figure 49, the combined parameter menu contains an instance of the signal generator and oscilloscope parameter menus, with a few small changes. The graph view box contains three options as opposed to the signal generators two, and there is an addition of a PCB trace between sections.

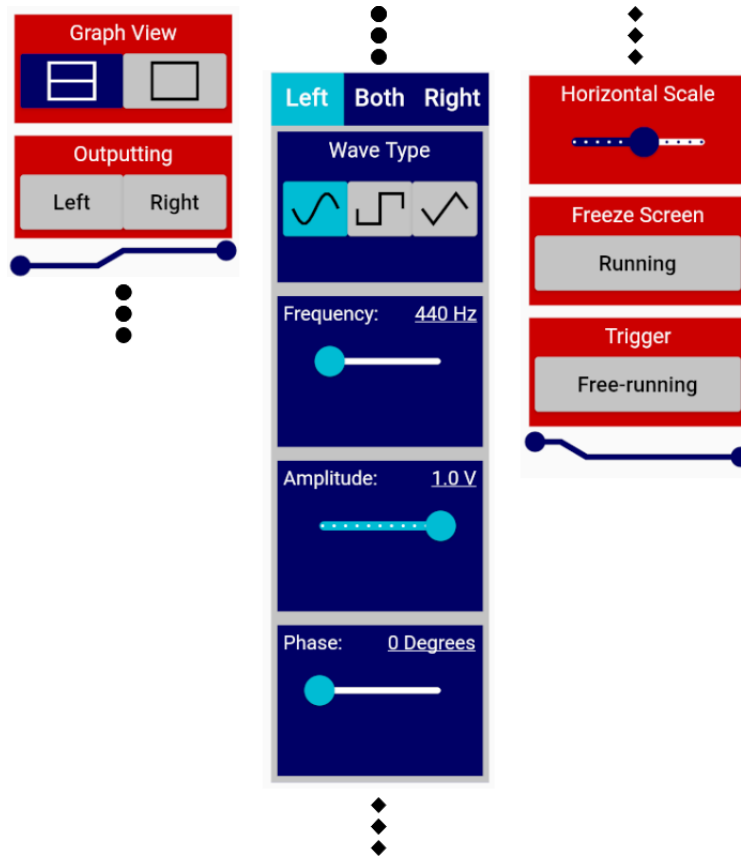


Figure 49: Combined parameter menu (symbols representing a vertical continuation between sections)

### 4.8.4 Graphing (Gavin)

The graphing section is aimed at visualising the signals being generated and received by the application. The page can either be used in signal generator mode, oscilloscope mode, or combined mode. There are two generated signals, a single incoming signal, or all three signals that need to be displayed simultaneously. The priority for this page is flexibility for the user to configure the graphing layouts.

Both the signal generator and combined modes show multiple signals, and so when discussing the design, it was decided that the graph layouts needed to allow the user to configure the graph layout to best suit their learning experience. This page can run experiments that demonstrate concepts such as superposition, which can be challenging to comprehend and visualise. Some users might better visualise concepts with multiple signals on a single graph, while others may find it easier with signals on separate graphs.

The “Graph View” box in the parameter menu is used to control the graph layouts. The selector icon indicates the graph layout (seen on the left side of Figure 50). When a layout is selected, the icon changes from black to white and the box is highlighted in blue. The graph views for the signal generator mode can be seen in Figure 50. The demonstration of the combined mode can be seen in Figure 51.



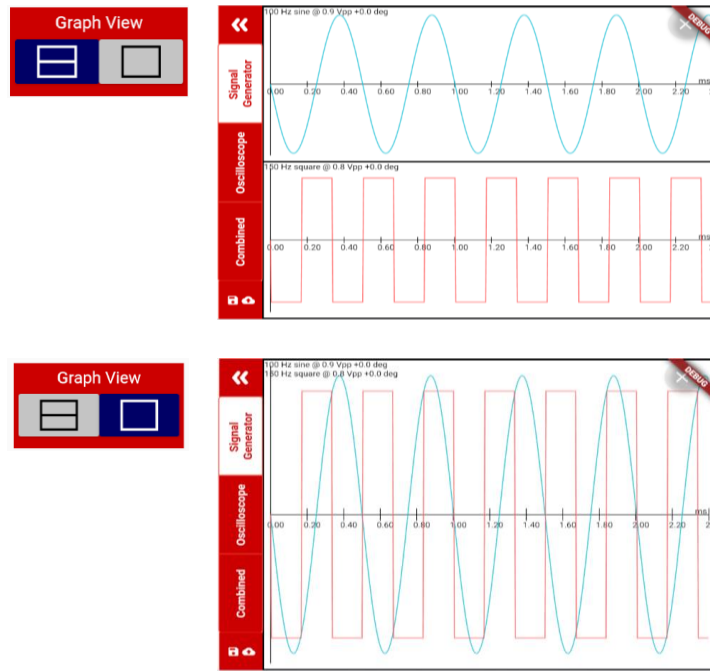


Figure 50: Signal generator mode showing (Top) two graphs (Bottom) one graph

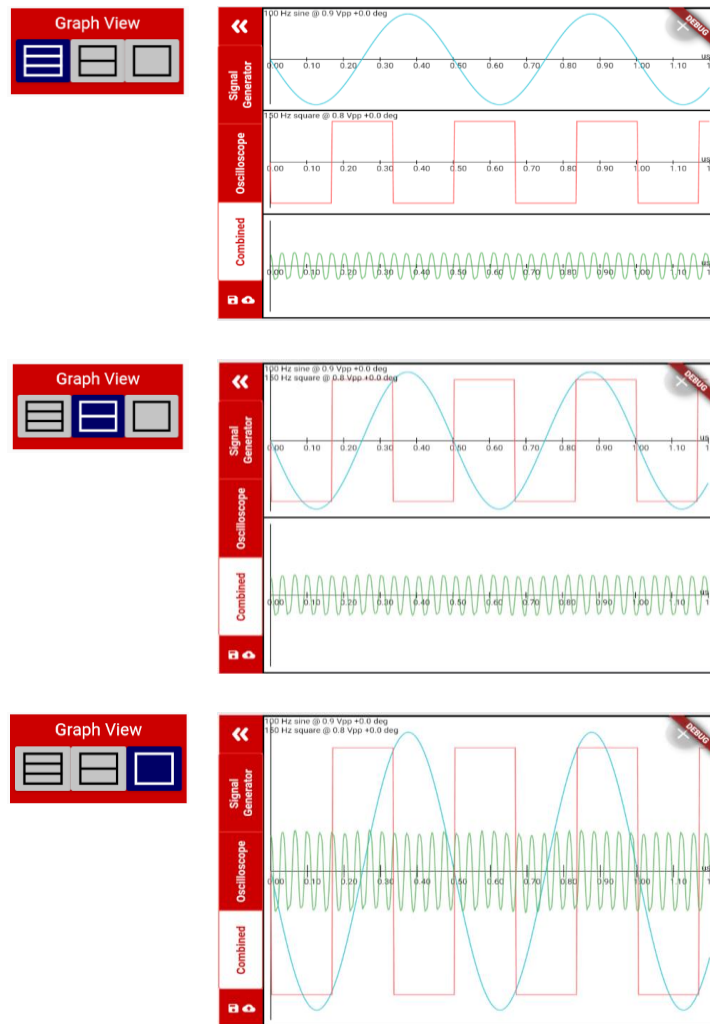


Figure 51: Combined mode showing (Top) three graphs (Middle) two graphs (Bottom) one graph

### 4.8.4.1 Graph Signal Labels (Gavin)

Information on the wave parameters is placed at the top left of the graph area for any graph showing a signal generated wave. This information is shown in a way as not to hide any other important information on the graph. There is also an indication that the signal is on, as there is an alert when the signal is switched off, and when multiple signals are being shown on the same graph, there is a parameter text for each signal, all of these are shown in Figure 52.

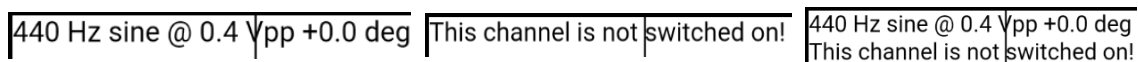


Figure 52: Text parameters displayed at the top left of the graph for (Left) a single signal on (Middle) a single signal off (Right) multiple signals (one on, one off)

### 4.8.4.2 Graph Style Configuration (Gavin)

As previously discussed in 4.7, the user can also choose some settings for the aesthetics of the graph. Some users may be more familiar with a traditional graph axis, while others may prefer an oscilloscope axis. Dark and light mode was also included because it is becoming more popular for applications and adds more choice for users. The default settings are light mode, traditional graph axis, which was most likely the familiar case for A-level physics students. The matrix of all combinations of theming are shown in Table 4

Table 4: Matrix of the four combinations of theme and grid style available

	<i>Dark mode enabled</i>	<i>Dark mode disabled</i>
<i>Oscilloscope axis enabled</i>		
<i>Oscilloscope axis disabled</i>		

### 4.8.4.3 Graph Backend (Eamonn)

The graphing on the practical page has several different operation modes, relating to the overall layout of the page. There are three primary considerations about the operation of the graph, the scale, the triggering and how many samples to store and display.

The scale requires considering the mode of operation, as well as the intended use of that mode. For example, if the page is in signal generator mode, the scale is automatically calculated. The formula used to determine the scale index to select is shown in Figure 53.

This allows the signal generator graph to automatically scale to see the full detail of the waves without any additional user input.

```
lScale = (log(model.lFrequency) / log(10)).floor();
rScale = (log(model.rFrequency) / log(10)).floor();
avgScale = (lScale + rScale + 4).toDouble();
avgScale = zoomScales[avgScale.toInt()];
```

Figure 53: Code for finding the scale between two frequencies of signal generation

However, if the user is in either oscilloscope or combined screen mode, the scale is based on the oscilloscope scale, set manually by the user. The scale values are chosen by the notch the user has selected on the slider, relating to the values found in Table 3. The number of points drawn for the signal generated graphs and the oscilloscope graphs are fixed to keep performance high. The scale determines the steps used to generate the height of the points for the signal generated points. It also determines the number (and steps) of the input buffer to be drawn on the oscilloscope. The detail of how the buffers are drawn will be discussed later in this section. Due to the combined page showing the scale set by the user, and there being a fixed number of points, the details of the signal generated graphs can be lost. It is assumed that the user will notice that no useful information is gained from an incorrect scale and will adjust until correct operation is found. An example of this can be seen in Figure 54 (Note the lack of clarity on the left image).

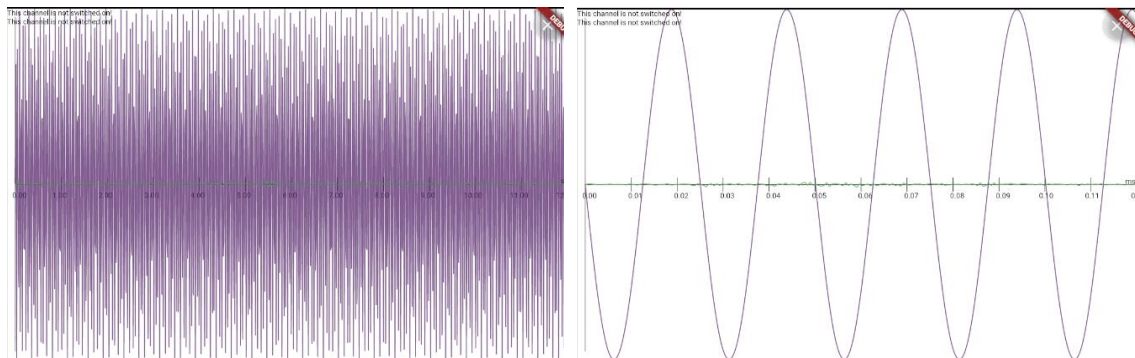


Figure 54: A 20 Hz sine wave is shown on the combined screen using (Left) a total time of 12 s (Right) a total time of 0.12 ms

The number of buffers stored for the graph page is calculated based on the length of the buffers. It assumes a sample rate of 44.1 kHz, and so based on this will store 12 seconds worth of buffers (or  $44100 * 12 = 529200$  samples). Doing this allows the user to scale in and out whilst retaining all the historical information to be displayed. As stated previously, the oscilloscope only draws a set number of points, as 529200 points on-screen would show no useful information. Therefore, the oscilloscope takes a sample every step based on the scale. This means that at a full display of 12 seconds, the steps are every 2000 samples. At the smallest scale of 0.006 seconds, every point is drawn. Both of these calculations result in 264 points being drawn.

The oscilloscope also needed to display lined up with the signal generated signals on the combined screen. To do this, triggering needed to be implemented. The visible points (chosen from the scale) as checked against the previous point, and if there is a negative incline through zero, then the sample index is added to an array. If the triggering is active, the median value from the array is selected as the offset. This triggering assumes a simple

periodic wave with only one negative edge that crosses zero in each period. This helps to keep the signal visible and aligns the oscilloscope with the signal generator graphs. A demonstration of this in operation can be seen in Figure 55.

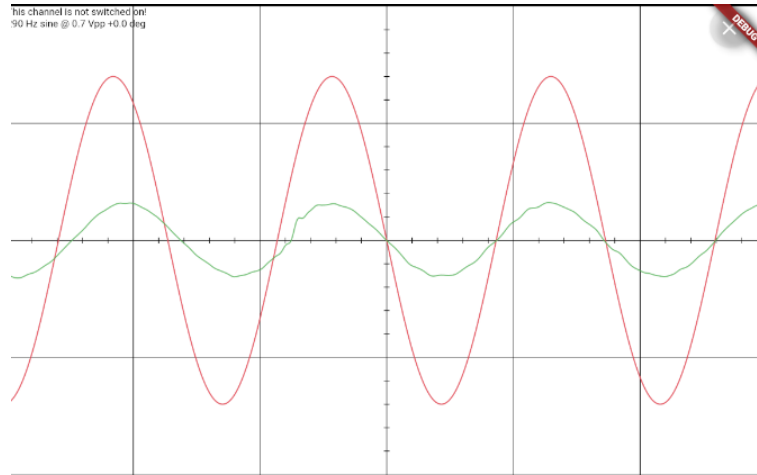


Figure 55: A demonstration of the triggering working for a signal generated wave

#### 4.8.5 Save/Load Slots (Foivos)

Enabling the persistent storage of the configuration of the practical page parameters is critical to ensure that users do not lose information when continuously using the application. An additional use of save and load configurations is the ability to return to a configuration of an experiment if the user could not complete it within a single session. The “shared\_preferences” Flutter plugin was used for the persistent storage of data. “Shared\_preferences” uses `NsUserDefaults` on iOS and `SharedPreferences` on Android to store simple data on the local storage for the device.

Only the configuration parameters were stored when storing parameters, meaning no real-time information was recorded (such as the oscilloscope buffers). This included the frequency, amplitude, offset, wave type, and outputting variables for the signal generator. For the oscilloscope, this included the triggering and frozen status.

Once the save/load button on the sidebar is pressed, an animated popup is displayed, as shown in Figure 56. The popup contains three different slots with the buttons to save, load, delete or get more information.

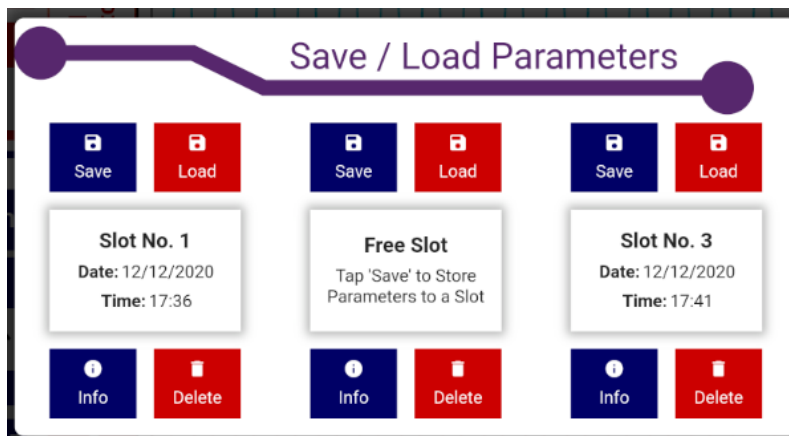


Figure 56: Save/load parameters popup with middle slot free

Free slots are also indicated and can be used so that a previous slot is not overwritten. As shown in Figure 56, these slots indicate if they have information stored. It is also worth mentioning that if a particular slot has not been saved to, the load, delete, and information buttons act as placeholders serving no function. Additionally, whenever a slot is saved, the date it was created, and its timestamp are displayed, allowing the user to have some defining property directly available without clicking the information button. The full list of stored parameters is seen if the information button is pressed.

#### 4.8.5.1 Saving to a Slot (Foivos)

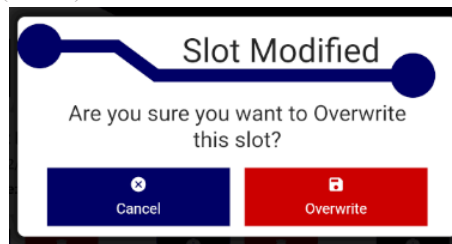


Figure 57: Confirmation dialogue for saving to a slot

If the user desires to save parameters on an empty slot, they can press the save button. If the user wants to save on a slot that already has data, a confirmation dialogue will show, as shown by Figure 57. Only on confirming the choice to save, will the slot be overwritten.

#### 4.8.5.2 Deleting a Slot (Foivos)

The user can delete a slot if they think it is no longer needed. Once the delete button is pressed, the user will be given a confirmation dialogue (same widget as the overwriting pop up) to delete. The slot is only deleted upon confirmation in this popup. This is shown in Figure 58.

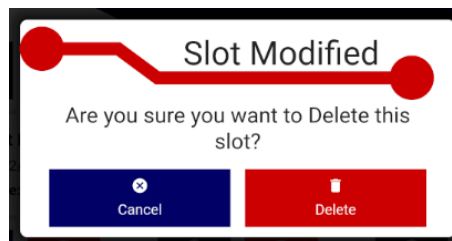


Figure 58: Confirmation dialogue for deleting a slot

#### 4.8.5.3 Information Popup (Foivos)

The user can view more information on a specific slot by selecting the information button on a slot with information. This will show a popup displaying three separate tables of information, as shown in Figure 59.

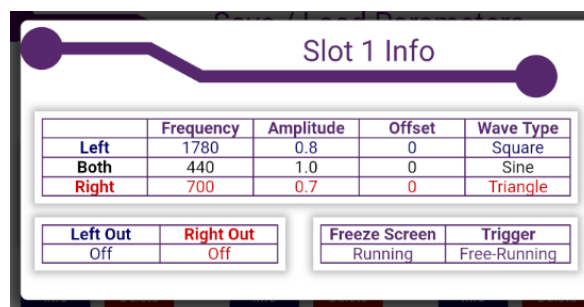


Figure 59: Information popup for the save/load slots

The first table contains all parameters relating to the signal generator. This includes the frequency, amplitude, offset, and wave type for each channel. The bottom-left table shows if the signals were outputting. The final table displays the two parameters for the oscilloscope, triggering and frozen.

#### 4.8.5.4 Default Configuration (Foivos)

When a user is continuously within the application, the persistent storage of their previous configuration on the practical page is an important feature. This prevents the user from exiting the practical page and losing their configuration for their running experiment. Therefore, a default slot automatically stores previous configurations when the user exits the practical page. This slot is restored when the user re-enters the practical page. This was implemented to allow the user to explore other areas of the application, such as the experiment instructions, without losing their configuration.

The default slot is deleted once the application is terminated. This slot is deleted when the application is launched, meaning if the application is running in the background and resumes, the information will not be deleted.

## 4.9 Application Navigation (Leonardo)

On Android devices, on both the Music Mixer and Logic and Arithmetic screens, if the user presses the back button twice, a confirmation popup appears, shown in Figure 60. If the user presses no, the popup closes without any action. If the user presses yes, the user is redirected to the home screen. This implementation can only be used on Android as iOS devices do not have back buttons. This is why there is also a home icon in the AppBar that will return users to the home screen. This implementation is possible by using a WillPopScope widget, which calls a function when the user pressed the back button. The function checks how many times the button has been pressed before activating the popup. The design of the popup is similar to the save and delete slot popup in the practical page.



Figure 60: Go home popup for kit pages

Android can use the navigation buttons to exit the practical page, while iOS users do not have these buttons available. As a result, a floating exit button was implemented in the top right of the practical page to allow iOS users to exit the page. This allows users of either platform a quick way to leave the page by pressing this. This additional implementation is still available for Android users, and the intention was to keep the user experience on both platforms identical. The exit button can be seen in Figure 61.



Figure 61: Exit button on the practical page

## 4.10 Integration (Eamonn)

Throughout the project, the individual implementations were developed on their respective branches on the project GitHub. Throughout the project, these separate branches were used to merge and branch to avoid any combability issues integrating the functionality. This integration also required some manual input due to merge errors or the combination of different variables.

The most challenging integration process arrived after the initial development of native features since these were developed on independent applications to aid in the speed of debugging. After combining the individual components, each team member continued to use their separate branches relating to a different functional implementation.

During integrating a new functional component, several critical reviews had to be made. The first was the ability of the code to be read in combination with the already existing code. This process was made easier by the user interface for merge conflicts on Visual Studio Code which visually shows the differences and allows a simple selection of which to keep (compared to command line merge conflict resolution).

When choosing the merge resolutions, the most significant consideration of the integration was the consistency of naming variables. Since the project might be continued professionally, it was essential to ensure the naming of variables were clear, so the code was readable and consistent. Luckily, Flutter has some conventions for the naming of classes compared to variables, and the ability to name private variables with an underscore prepending the variable name to indicate the privacy. These rules are sometimes missed when there is a quick debugging and developing cycle on the branch, so it was essential to include these consistency checks at the time of integration.





## 5 Testing (Eamonn)

### 5.1 Methodology (Eamonn)

Throughout the project, rudimentary testing was completed during the development phase of the application. This allowed for rapid iterations and ensured that any bugs were found and resolved as soon as possible. However, these preliminary tests did not have a high accuracy level to determine the deterministic behaviour of the application. At the end of the project, the application was thoroughly tested within a laboratory to ensure a known full performance characteristic.

The building 16 laboratory on the campus of the University of Southampton was restricted during testing due to Covid-19 restrictions. Despite this, it was still possible to produce a full test suite to determine the performance difference on several devices. The tests were performed in the same manner, independent of the team member carrying out the test was crucial.

During testing, the laboratory oscilloscope was connected to the laboratory computer with software to capture every result. Similarly, the oscilloscope was configured with the current test measurements, and that test alone, ensuring the image contained only relevant data. Any investigations that did not have a specific measurement in mind recorded as many measurement characteristics as possible (relating to amplitude, frequency and relative phase). Figure 62 shows the full testing set up (including a diagram of the oscilloscope connections to the board).

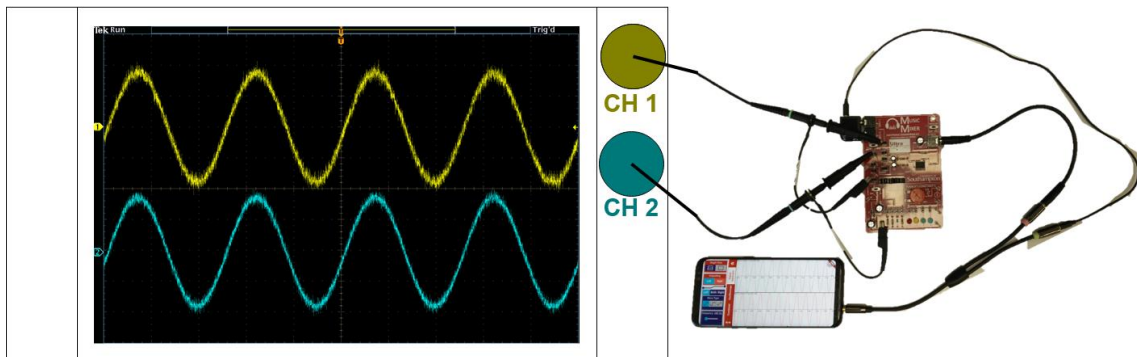


Figure 62: Testing configuration using a Samsung S8

The testing involved trying to determine several factors about the performance of the application. The first was how accurate the frequency behaved compared to the expected. This process was repeated for the relative phase output, and then the amplitude. It was also crucial to discover if different devices had different maximum amplitudes, so the amplitude was tested in a matrix to ensure that each absolute value was determined. Beyond these set values, a comparison of each devices performances at different frequency ranges for certain wave types was tested, and then a test on adding different waves of different frequencies and phases to determine correct behaviour.

## 5.2 Android Results (Eamonn)

### 5.2.1 Samsung S8 Using an Auxiliary Cable (Eamonn)

Samsung S8 testing shows a strong performance of the application. The frequency tests show high accuracy reporting an average error of just 0.66%. The highest recorded error was 1.54%, and the lowest was 0%. The results can be seen in Figure 63.

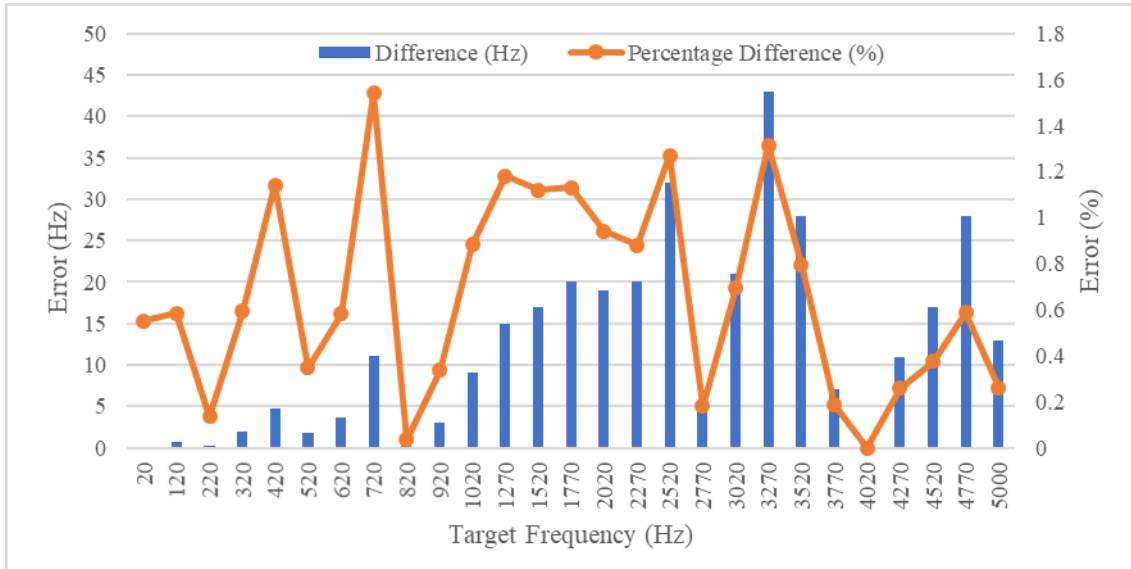


Figure 63: Graph showing the target frequency compared to the recorded frequency using the auxiliary cable on a Samsung S8

Similarly, the results for the phase difference support the argument that the application has high accuracy. The differences recorded range from an error from 0.09° to 2.04°. The average error is only 0.85%. The results can be seen in Figure 64.

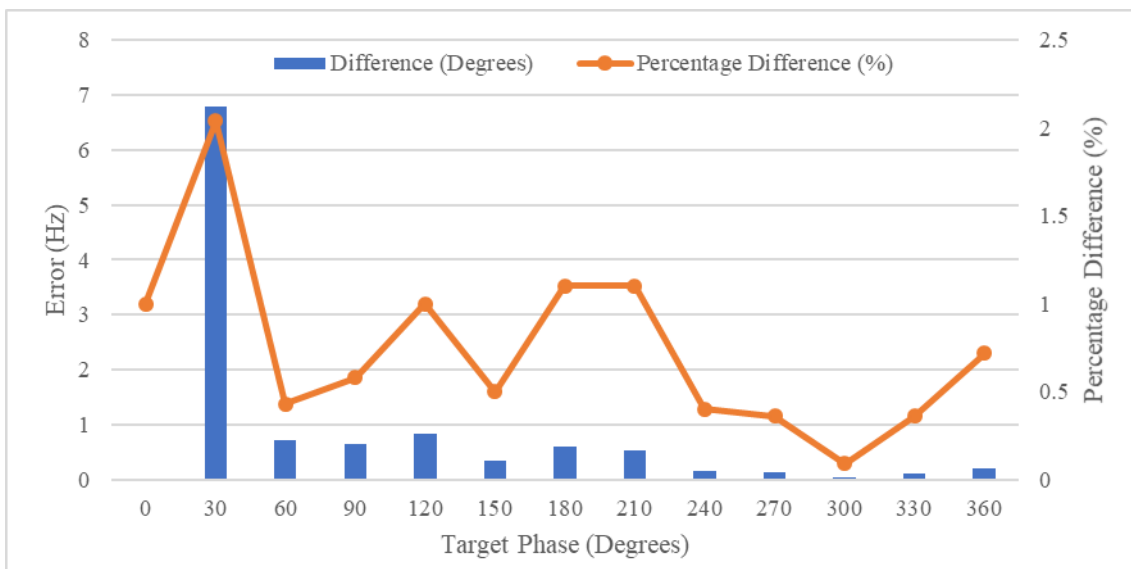


Figure 64: Graph showing the target phase compared to the recorded phase using the auxiliary cable on a Samsung S8

The amplitude testing for the Samsung S8 shows an exponential trend on the devices volume setting, with a linear trend using the amplitude slider included within the application. These results were in line with what was expected. If the results were

consistent with the rest of the tests completed on Android and iOS, the amplitude axis in the application could be labelled. The amplitude for the Samsung S8 had a maximum amplitude recorded by the laboratory oscilloscope of 2.24 V. The minimum recorded amplitude was 20.8 mV. However, this is due to general noise on the oscilloscope, as no measurement could be 0 V in reality. Figure 65 shows the amplitude values at different “notches” of the phone volume.

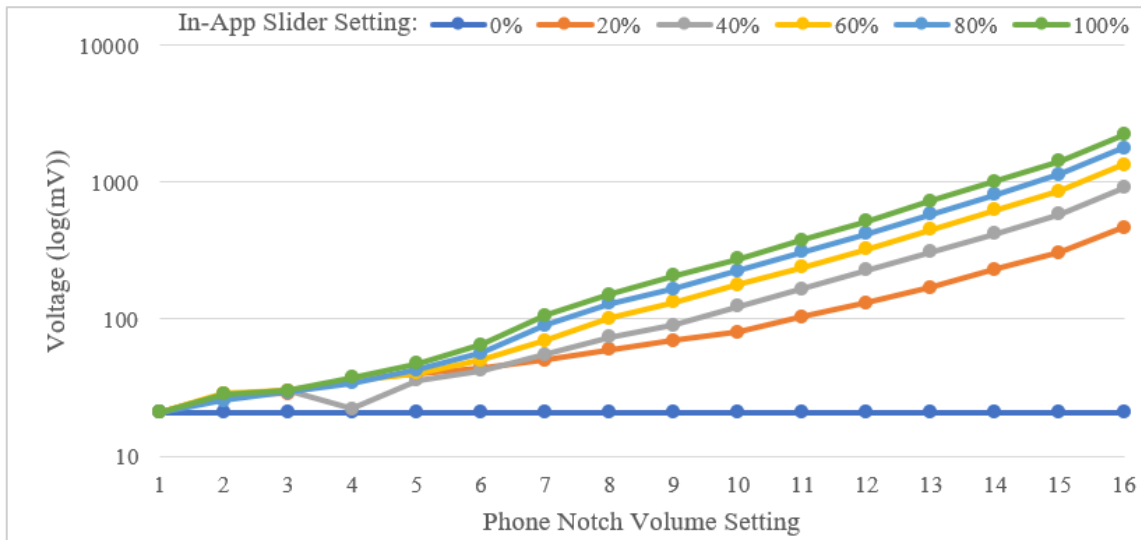


Figure 65: Graphing showing recorded amplitudes for different phone and slider volumes on a Samsung S8

### 5.2.2 OnePlus One Using an Auxiliary Cable (Eamonn)

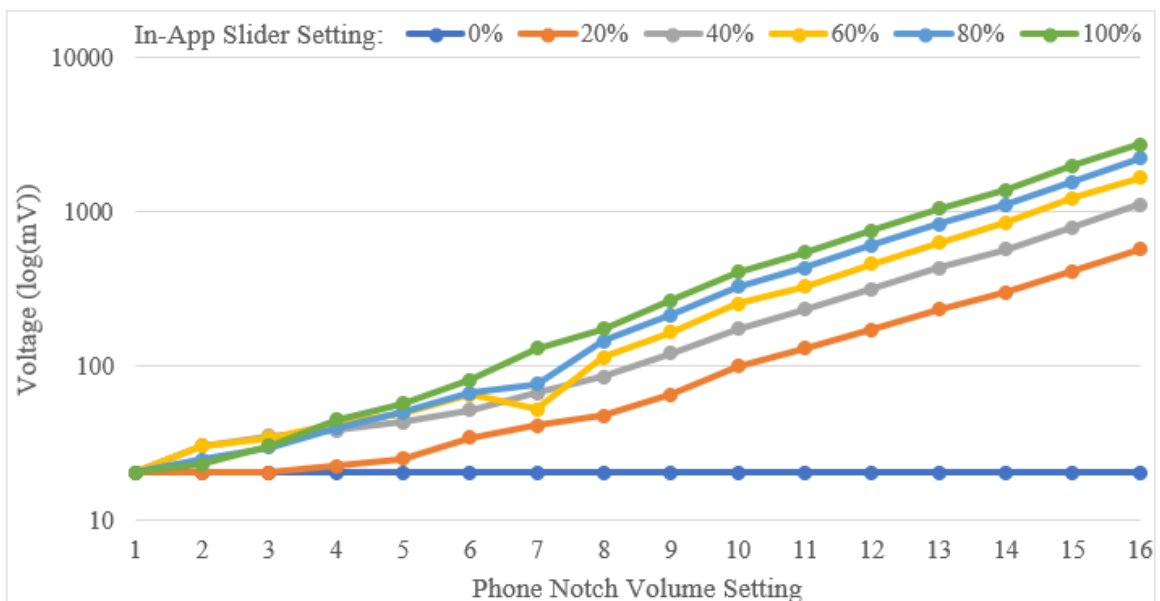


Figure 66: Graphing showing recorded amplitudes for different phone and slider volumes on a OnePlus One

The accuracy of the frequency and phase were tested to ensure similar characteristics. However, the measurements were not recorded because the frequency and phase are software limited. The maximum recorded amplitude was 2.76 V, and the minimum recorded amplitude was 20.4 mV (but similar to the Samsung S8, this is due to noise on the oscilloscope). These results are shown in Figure 66. These are different from the Samsung S8 result and support the decision not to place a scale within the application.

### 5.2.3 Samsung S8 Using Other Connections (Eamonn)

The Samsung S8 was used to test other connectors that could be used with the Music Mixer board. The first one to be tested was a Bluetooth dongle. It only had the capability to output audio, and therefore the ability to test the input was not possible.

The Bluetooth dongle had all the same characteristics as the auxiliary cable for outputting. The absolute accuracy for frequency and phase was not measured, as it was assumed that the accuracy would be software limited, and therefore could be determined by using the auxiliary cable measurements.

The difference between the Bluetooth dongle and the auxiliary cable was that the Bluetooth dongle had a DC offset applied. It also appeared that the absolute amplitude maximum was different from the auxiliary cable, reaching only 1.72 V through the Bluetooth dongle (with the DC offset removed).

Similarly, a USB-C-to-auxiliary cable was investigated, and the same conclusion was drawn. The amplitude was not the same absolute measurement, reaching a maximum of only 552 mV. This is supported in discussing the results found when testing the OnePlus One (section 5.2.2).

## 5.3 iOS Results (Eamonn)

iOS testing followed the same methodology as the Android testing, meaning the amplitude, phase and frequency were all tested for their accuracy and absolute values. Unfortunately, due to only having an iPhone XS available for testing, there was no possibility to test using an auxiliary cable. Only the lightning cable to auxiliary cable was tested on iOS.

The frequency of iOS was similar to Android, with a minor average inaccuracy of 0.14%. The maximum inaccuracy was 0.5% with a minimum of 0%. Based on these values, iOS is more accurate than Android. The full results are shown in Figure 67.

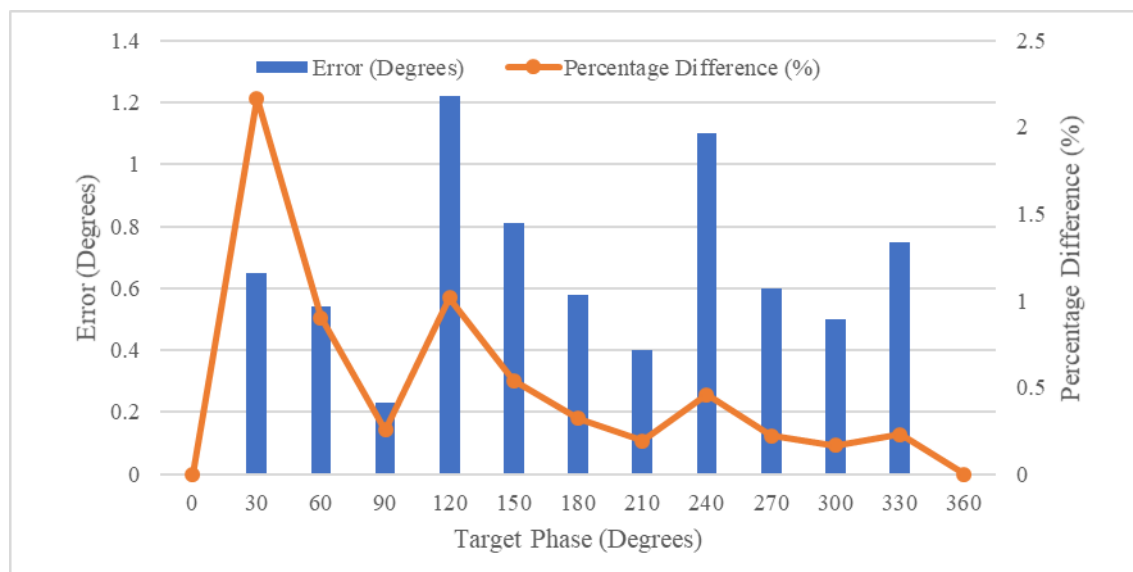


Figure 67: Graph showing the target frequency compared to the recorded frequency for using the lightning to auxiliary cable on an iPhone XS

The phase recorded on iOS uncovered a bug in the code. Each output thread was initialised at independent times; therefore, there was an offset on the phase difference between the two waves. This means the raw values have an offset of  $108.8^\circ$  in the following results. To allow for relevant analysis of the results, each recorded value had  $108.8^\circ$  subtracted from it. With these corrected results, the phase had an average inaccuracy of 0.49%, reaching a maximum of 2.16% and a minimum of 0%. The graph and values for the results can be seen in Figure 68.

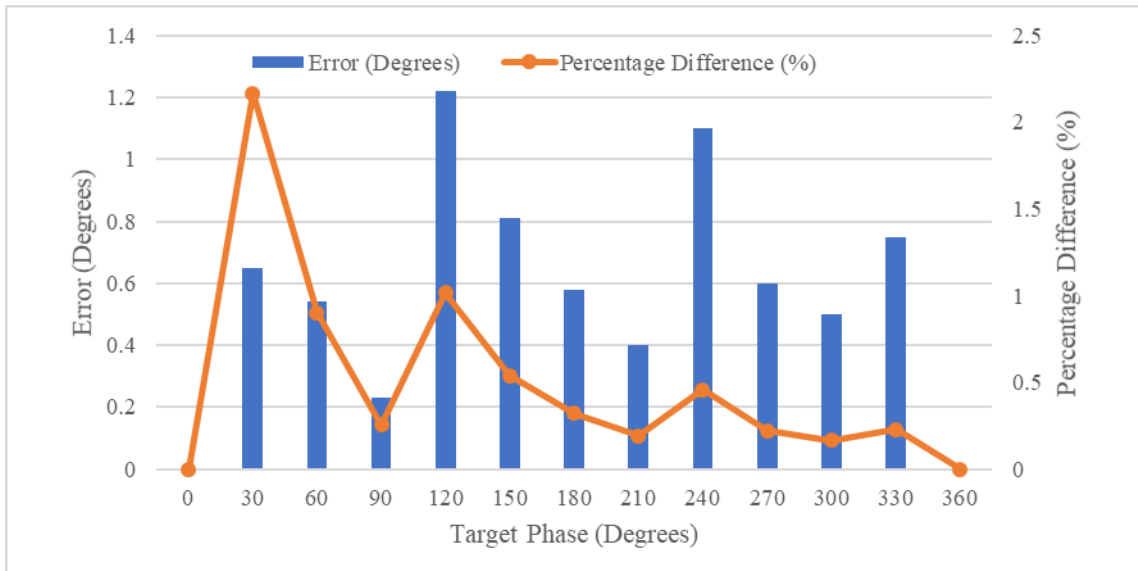


Figure 68: Graph showing the target phase compared to the recorded phase using the lightning to auxiliary cable on an iPhone XS

The amplitude shown on iOS is different in absolute values to those on Android. The maximum amplitude recorded was 2.72 V, and the minimum being 22 mV (due to noise). The results can be seen in Figure 69.

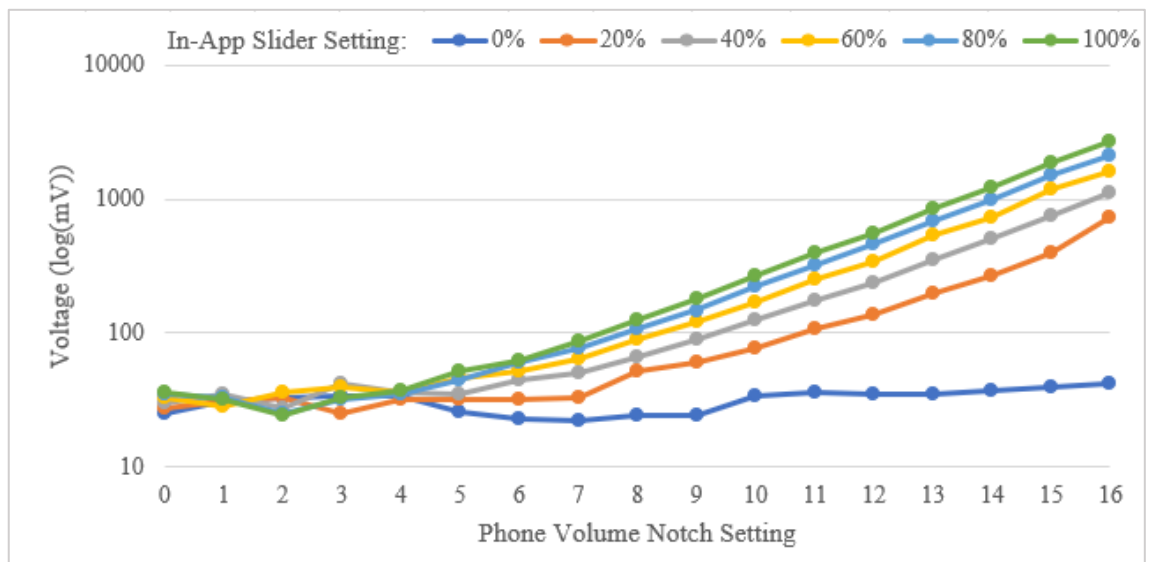


Figure 69: Graph showing recorded amplitudes for different phone and slider options on an iPhone XS

## 5.4 Other Notable Results (Eamonn)

### 5.4.1 Signal Generation (Eamonn)

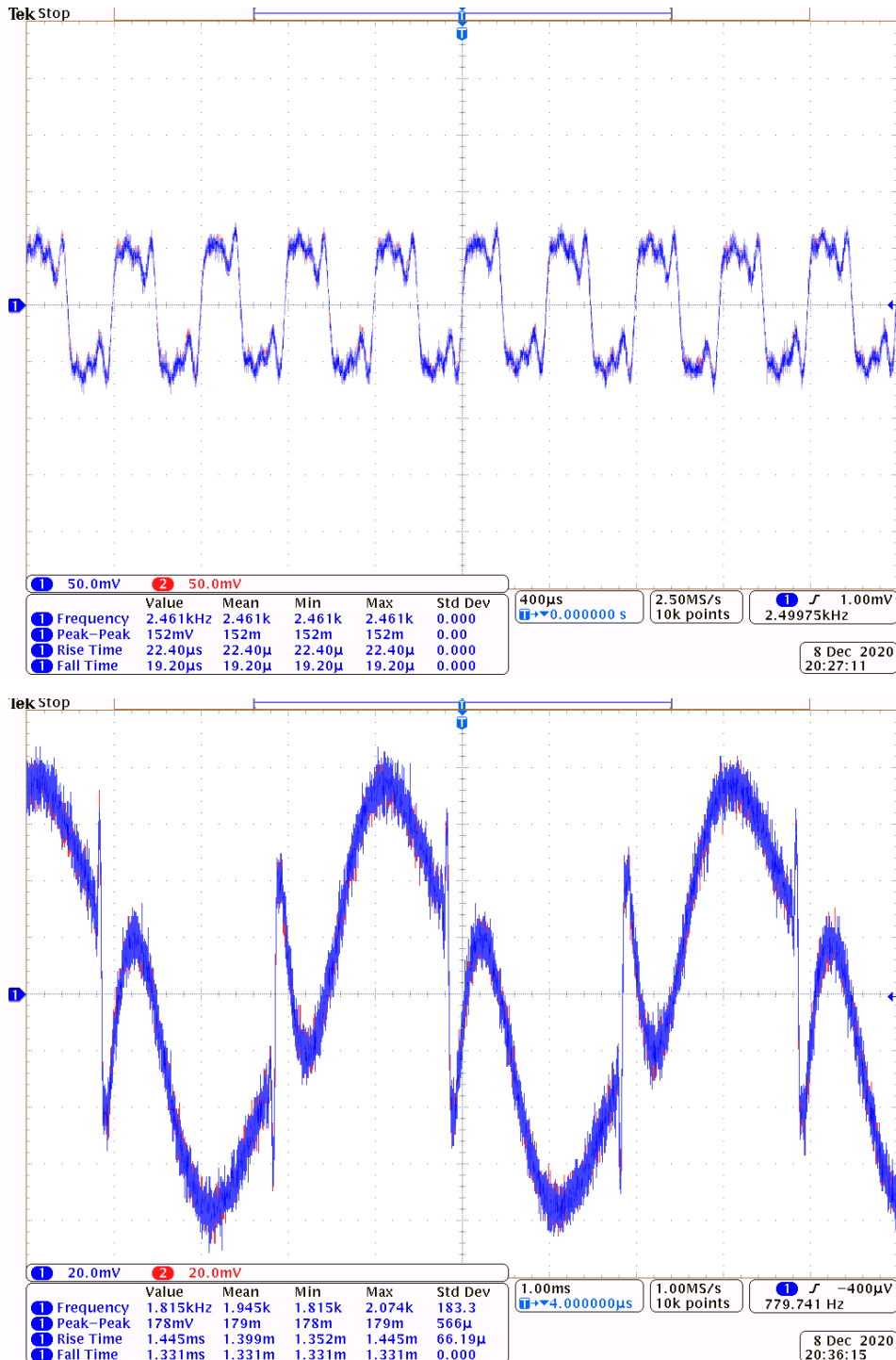


Figure 70: Triangle wave being generated by a OnePlus One(Top) correctly at 2500 Hz (Bottom) incorrectly at 250 Hz

Every device that was tested provided evidence that signal generation had different performances on different devices. The different devices could not produce the same quality signals that were not the fundamental sine waves at specific ranges. For example, when using a OnePlus One, the generation of a square wave was significantly less accurate than the Samsung S8 throughout the same range. In Figure 70 it is possible to see that at

a frequency of 250 Hz, the performance of a square wave is not adequate on the OnePlus One, whilst at a frequency of 2500 Hz, the OnePlus One can produce square waves with no problems.

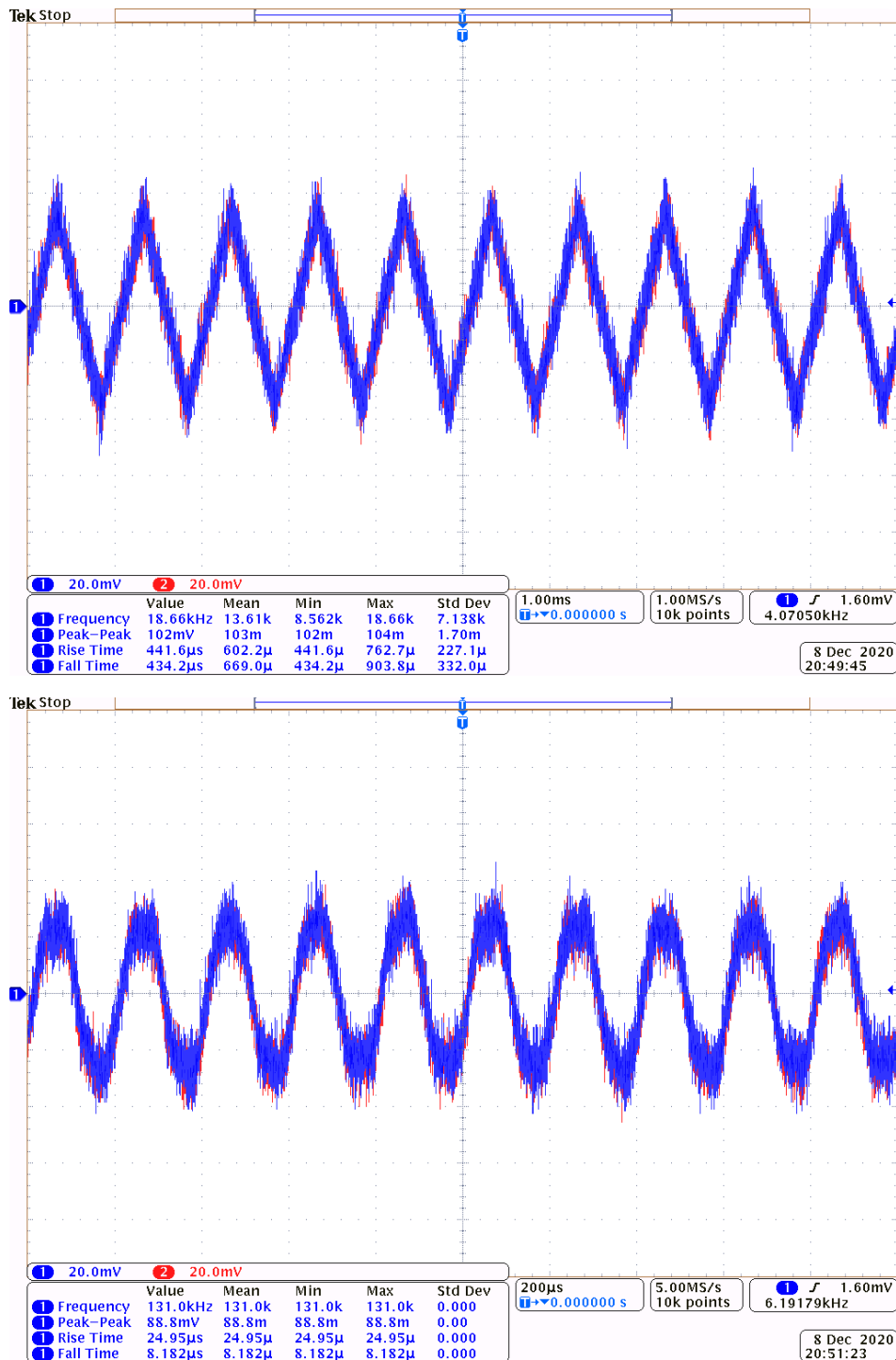


Figure 71: Triangle wave being generated by a Samsung S8 (Top) correctly at 2500 Hz (Bottom) incorrectly at 5000 Hz

Similarly, on the Samsung S8, the same variance per frequency is found, with the triangle waves being correct throughout most of the range, however at a frequency of 5000 Hz, the peaks and troughs seem to round (seen in Figure 71).

### 5.4.2 Microphone Detection and Processing (Eamonn)

During testing, both Android and iOS devices had problems connecting to the board in the first instance. It was quickly discovered that phones have specific hardware requirements to identify the auxiliary port lines to detect the hardware connected to it. This is done so that the phone knows how to handle the connection (i.e., an input or an output). The Android specification for detecting a microphone input to the phone is to have a  $100\Omega$  resistance across the line to the GND pin. The impedance also should be above  $5k\Omega$  to be detected as a line in [30]. iOS testing shows similar performance, requiring an in-line attenuator for the microphone to be detected. This attenuator has the resistances and impedances on the lines in order to prevent this issue.

### 5.4.3 Oscilloscope (Eamonn)

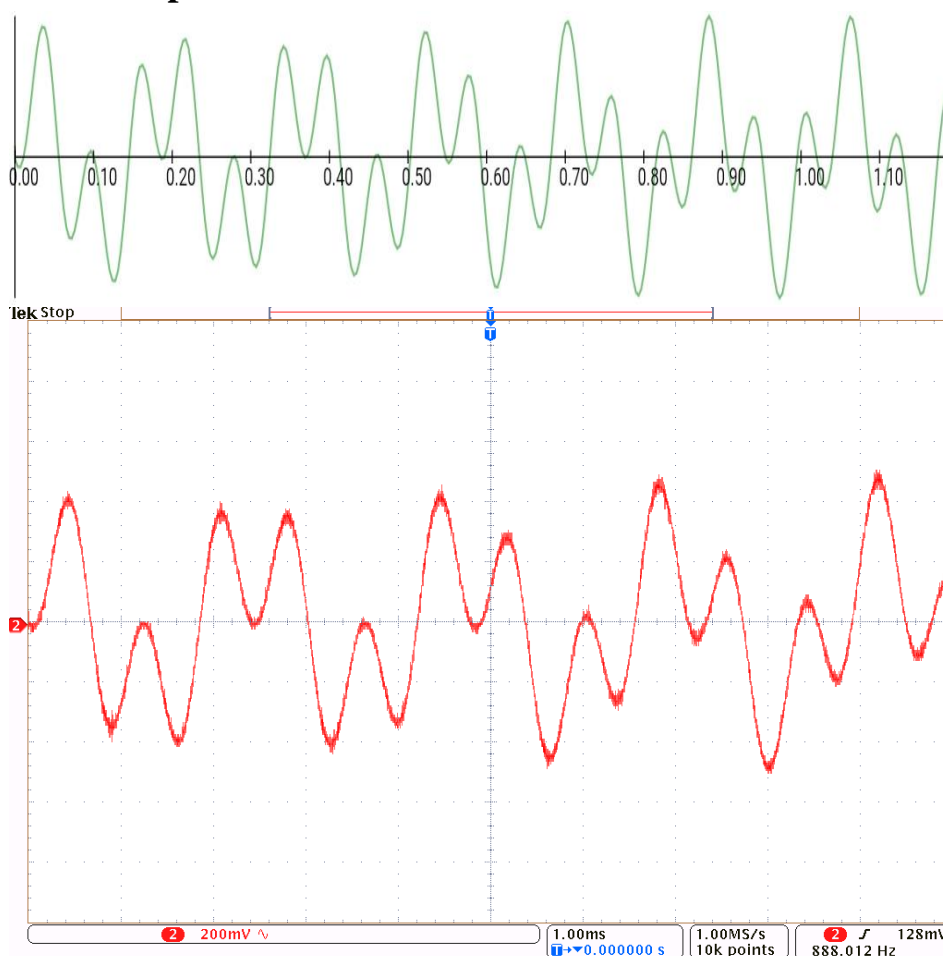


Figure 72: Constructive wave of 1260 Hz sine wave with a 440 Hz sine wave (Top) on the application (Bottom) on the laboratory oscilloscope

These inaccuracies found in the signal generator do not seem to impact the operation of the application relating to the oscilloscope and constructive waves. Several tests were performed showing the application, the waves produced, the wave input and the laboratory oscilloscope showing the same constructive wave. These can also be verified by eye as the constructive waves chosen were straightforward. The most notable was using a 1260 Hz sine wave combined with a 440 Hz sine wave, with no phase offset (seen in Figure 72). Another excellent example of performance is shown in Figure 73, where a 440 Hz sine wave was added to a 440 Hz triangle wave with a  $180^\circ$  phase difference.



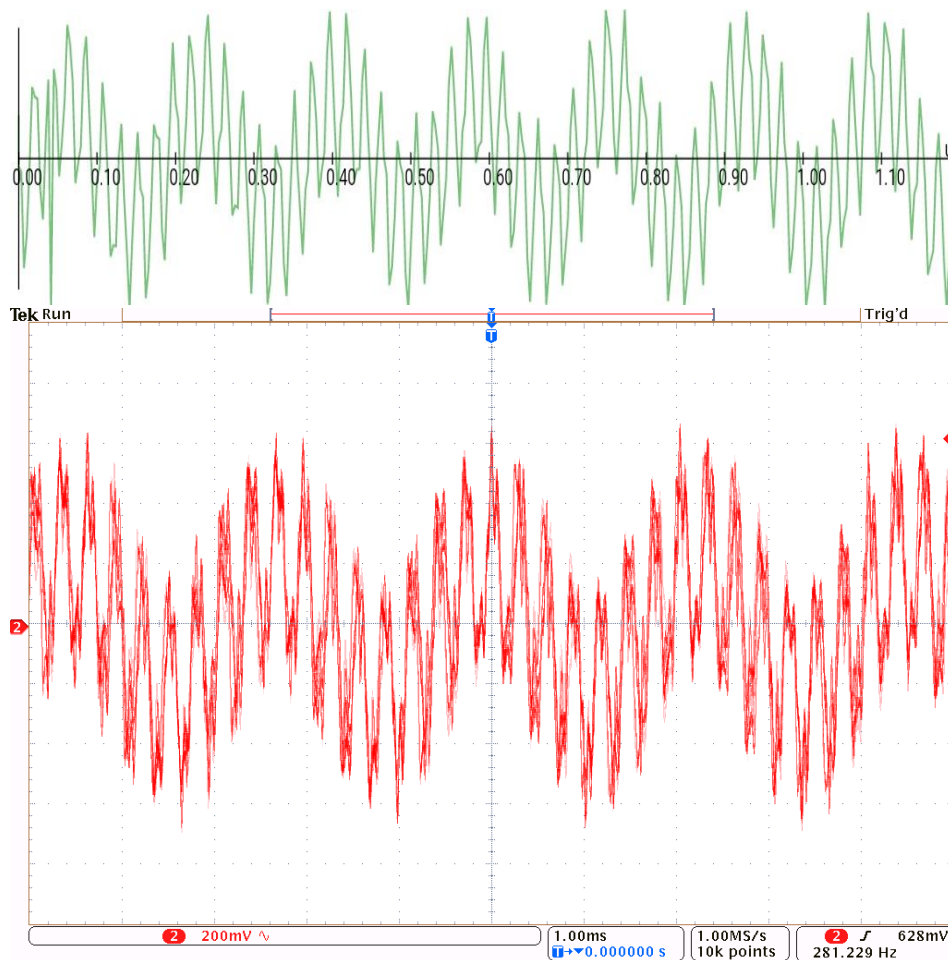


Figure 73: Constructive wave of 440 Hz sine wave with a 3430 Hz square wave with a 180° phase offset (Top) on the application (Bottom) on the laboratory oscilloscope

## 5.5 A-level Teachers Feedback (Eamonn)

### 5.5.1 First Survey (Eamonn)

The first questionnaire received 13 total responses. These responses gave a strong rationale for the project and raised some additional improvements that could be made to the hardware of the board. The additional comments were all out of the scope of this project; however, included statements with the general sentiment that more flexibility with the board would be beneficial (such as the ability to add a 5<sup>th</sup> LED to the Planck's constant section of the board quickly). A graphical display of the results can be seen in Figure 74.

### 5.5.2 Second Survey (Eamonn)

The second questionnaire received a total of eight responses relating to the understandability of the first iteration of the design of the project. Only 12.5% of teachers disagreed that the design was not cluttered on the oscilloscope page. Most of the responses indicated that the application was easy to understand, with no teachers thinking the combined application screen was cluttered, poorly thought out, or confusing to students. More specifically, 62.5% of teachers asked agreed that it was clear how the combined screen, would work, with a further 12.5% of teachers strongly agreeing. Similarly, five out of eight teachers agreed or strongly agreed that students would understand the design

with 50% agreeing or strongly agreeing that students would easily interact with the more complicated features. A graph of these results can be seen in Figure 75.

### **5.5.3 Third Survey** (Eamonn)

The final questionnaire was like the second questionnaire and was used to understand if the application had been designed to a good standard. Unfortunately, due to the timing of this survey and the complications with teachers work this year, there were only three total responses. This means concluding these results would be disingenuous as to make statements about the final design of the application. Nevertheless, the few results collected can be seen in Figure 76.



Figure 74: First survey responses

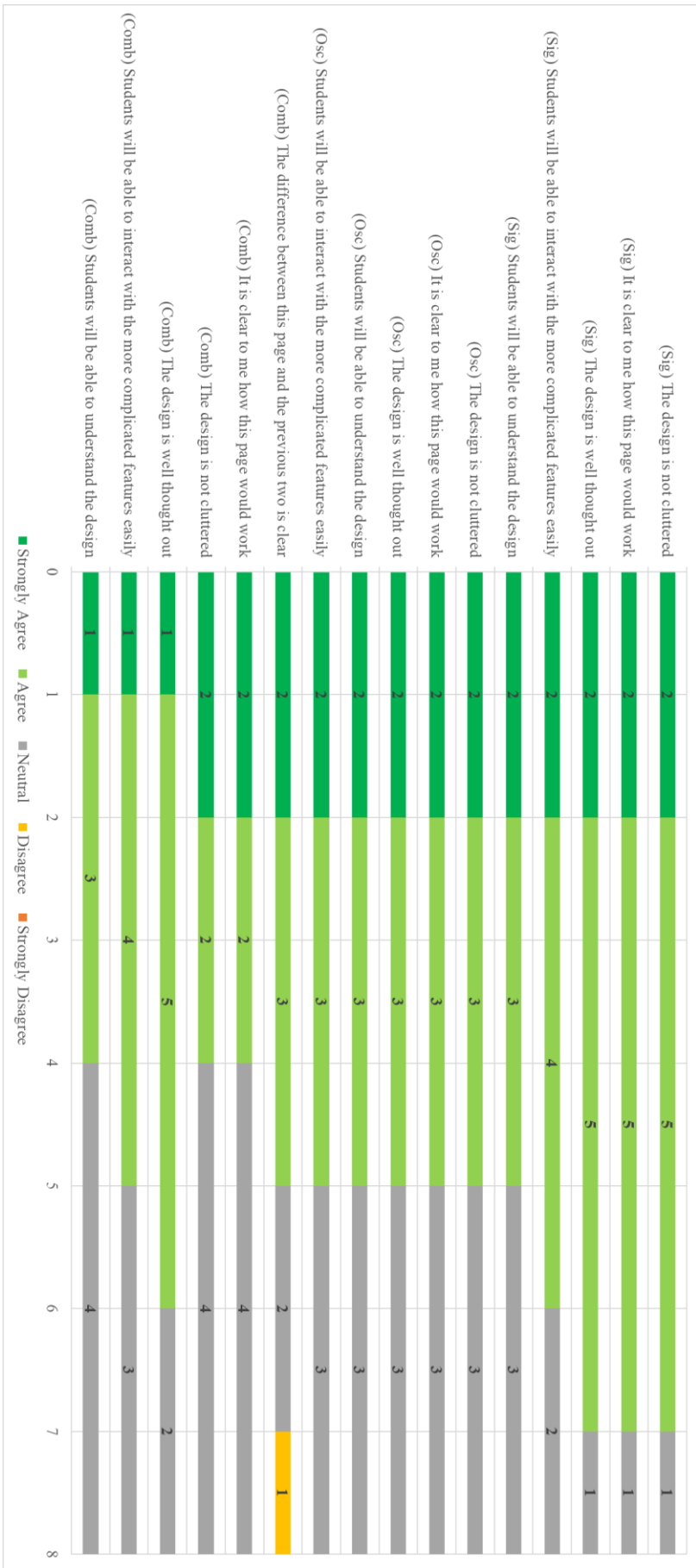


Figure 75: Second survey results

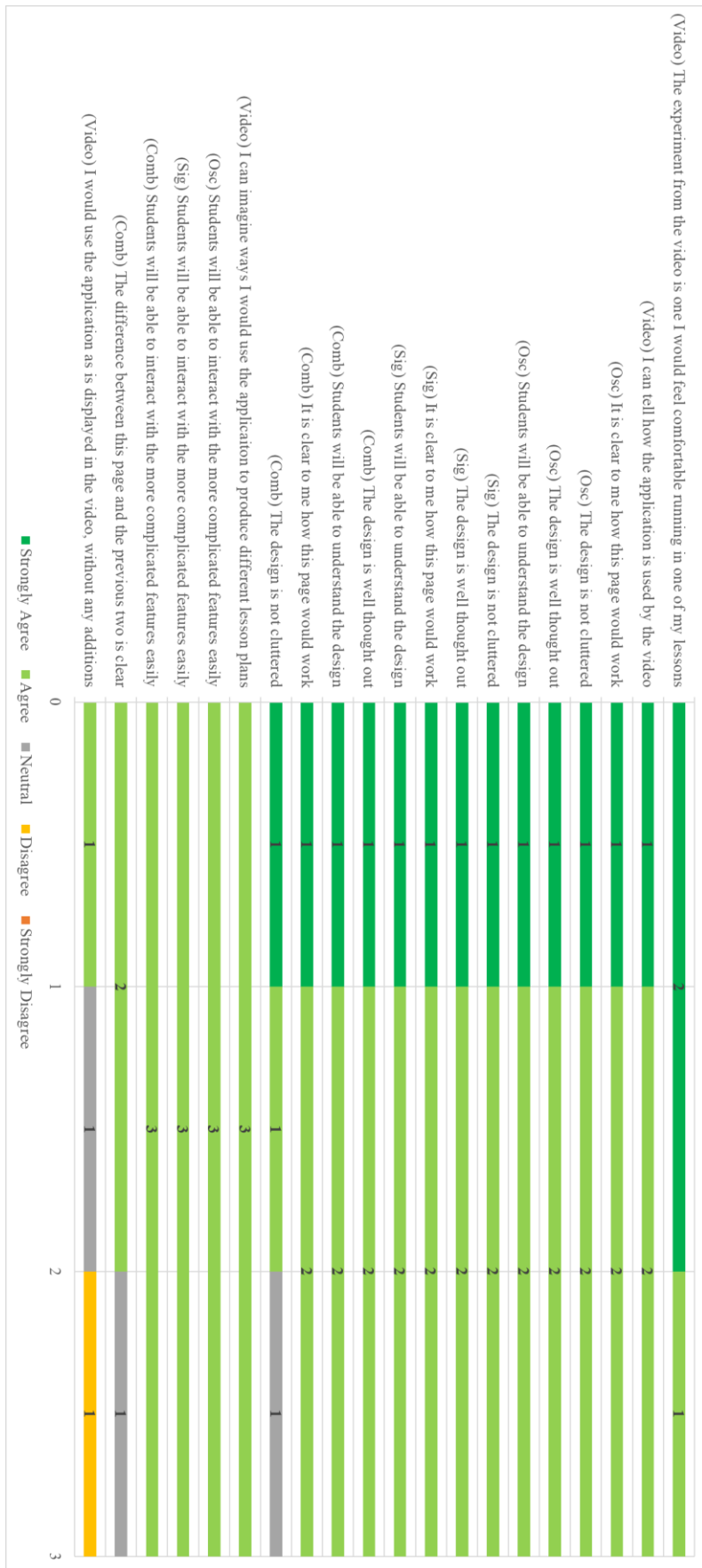


Figure 76: Third survey results



## **6 Final Outputs** (Gavin)

### **6.1 Application** (Eamonn)

Relating to the specification defined in 3, the project completed all the core requirements, including implementing a signal generator and oscilloscope at a sample rate of 44.1 kHz, save/load implementation and instructions for the material found on the Electronics Everywhere webpage. The application has also explored stretch features (mentioned in 3.1.2) with features such as the interactive image, information about the Logic and Arithmetic kit, and the Logic and Arithmetic instructions. Additionally, the other stretch features have been researched and are discussed in some of the handover resources discussed in 6.2.

### **6.2 Handover Resources** (Gavin)

In addition to producing the application, supporting resources were also created. These resources were created to aid a future team picking up the project where it was left off.

#### **6.2.1 Handover Document** (Gavin)

The handover document contains the information required for a new team to continue this project with all the insights from the current team. The document contains an introduction to the project, a list of implemented features, and known bugs. The handover document also discusses potential fixes to these bugs and a list of suggestions for further implementation (including explanations of how these might be achieved). Some of the content overlaps with section 9, with recommended features outlined and discussed. The discussion also includes ideas of the design and how the features might be implemented. The handover document can be found in appendix I.

#### **6.2.2 Instructional Video** (Gavin)

An instructional video was created that walks through the application, demonstrating the features [31]. This could potentially be used as part of the supporting content for the release of this application. This video is six minutes long, describing the features found in the application and their intended uses.

#### **6.2.3 Written Instructions** (Gavin)

To accompany the instructional video, a set of written instructions have been produced. These could be used as part of the supporting content for the release of this application after some branding has been added. An information leaflet is currently packaged with the Music Mixer kit, and a similar style could be applied to these application instructions. The document is an infographic with diagrams explaining the use of the application with images, which can be seen in appendix G.

#### **6.2.4 Music Mixer Laboratory Experiment Instructions** (Gavin)

The University of Southampton has produced some laboratory notes that contain experiments that can be run with the Music Mixer board. These were created before developing this application, and so have references to third-party applications. They also include references to external hardware, which can now be replaced with the Electronics Everywhere Application. Section “2.2 Potential Dividers (AC)” and additional section

“2.3 Music Mixing” have been edited to ensure that they can be read and used alongside the application. These new laboratory notes could either replace or accompany the existing laboratory notes for running experiments and can be seen in F.

### **6.2.5 Stock Images** (Gavin)

Within the project archive is a range of images taken of the Music Mixer kit working with the application in different scenarios. The idea of these images is to allow them to be used for promotional materials. They are used within the application. Some photos were taken with the purpose of promotion, so the aesthetics were prioritised, with other photos taken to demonstrate the use of the application. The photos promote the application and or the kits, either demonstrating use or as eye-catching content, and some highlights can be seen in appendix G.

## **6.3 Surveys** (Eamonn)

As will be discussed in 7.2, several surveys were sent to teachers to understand one of the end-users insights into the project throughout the project. These surveys came in the form of an initial survey to gauge the requirements of A-level physics teachers. In the middle of the project, a second questionnaire was sent, asking for feedback on the intermediate design of the project, and if it was good. Finally, a third survey was sent, which assessed how the final product was viewed by teachers.

## **6.4 Testing Documents** (Eamonn)

At the end of the project, a thorough test suite was run on the project (previously discussed in 5). These results were recorded in an excel document, and images of each result were saved to the project archive. These results helped to inform the characteristics of the application running, and have been put in the project archive so that the team following can use the results in their design decisions.



## **7 Project Management** (Eamonn)

### **7.1 Division of Responsibilities and Project Planning** (Eamonn)

#### **7.1.1 Initial Plan** (Eamonn)

As discussed in 3, the initial specification was developed from the brief and discussions with the external partner. From this specification, the initial division of tasks was devised. At the start of the project, the duties were split into different functional implementations that could be completed independently.

These independent tasks were then made into a Gantt chart. The most crucial components were placed at the beginning of the project to avoid a problem in development from causing cascading issues. The natural division of time in the project aligned with the academic hand-ins, involving three time periods named sprints.

The first sprint started with the academic year (5<sup>th</sup> October 2020) and lasted three weeks until the first presentation deadline (23<sup>rd</sup> October 2020). This sprint focused on the project brief, background research and development of the native implementations which handled the input and output interface on the different platforms.

The second sprint came after the first presentation (23<sup>rd</sup> October 2020) and lasted four weeks until second presentation (20<sup>th</sup> November 2020), involved integrating the individual native components to work together on one application and ensure a common GUI theming use throughout the project.

The third and final sprint was between the second presentation (20<sup>th</sup> November 2020) and the Christmas holidays (11<sup>th</sup> December 2020), lasting three weeks. This focused on bringing together the technical project, intending to combine all the user interface discussed as either stretch goals or additional features beyond the core.

These sprints were placed into an initial Gantt chart shown in Figure 77.

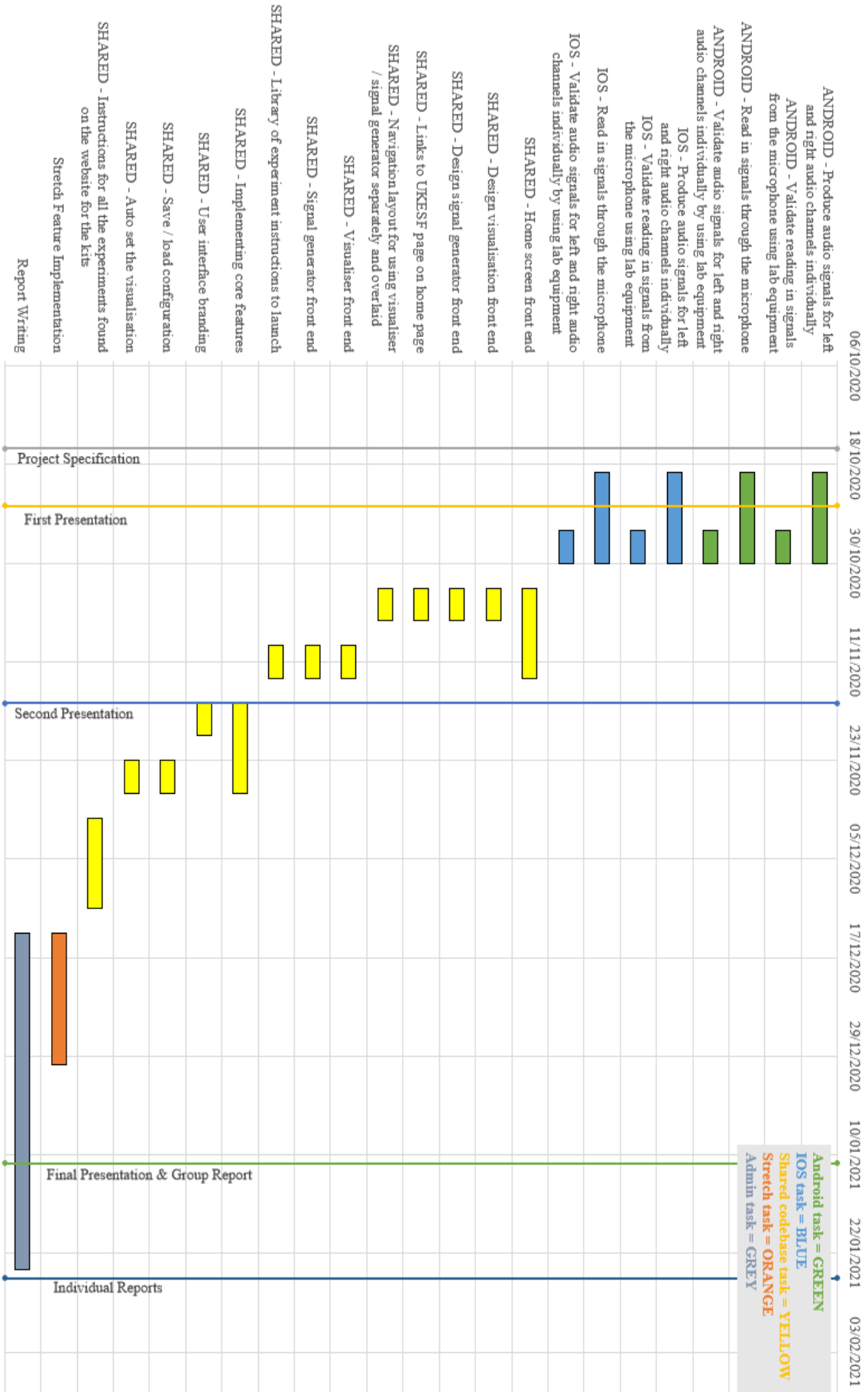


Figure 77: Initial Gantt chart with the sprints labelled

Since the project naturally had these divisions of time, it was decided that the ownership of each task would be agreed at the start of each sprint. If there were issues, they could be raised in the Weekly Operational Review (WOR), which occurred every Monday at 12:00-1:00 PM.

### 7.1.2 Initial Division of Roles (Eamonn)

During the first team meeting, it was clear that a team structure was required. As a result, it was decided that Eamonn Trim would be the project manager. The division of roles appeared to be quite natural for the first sprint, four group members being on the Electrical and Electronic Engineering degree, and the remaining member being on the Computer Science degree.

The project manager role included everything from handling the administration of meetings, contacting the client and supervisor, creating the Gantt charts, ensuring the team stayed on track, and managing the team members. As a result, there were no roles given for secretary or financial manager, as these were to be handled by the project manager, who agreed the administrative tasks would take precedent over the technical ones.

For the first sprint, the plan was the team members doing engineering would work on the hardware interface, whilst the computer scientist would work on the start of the software side of the project, including the basics of the user interface. The second sprint focused on designing and implementing the core user interface of the application. As a result, all the team members were expected to input the direction the user interface should take. After this core user interface scaffold, the final sprint was dedicated to adding the missing functionality, such as user experience features on the practical page, and adding an interactive image page to engage students in some background knowledge. These final tasks were assigned based on the strength and weaknesses matrix (discussed further in 7.5.1) when the team members reached the final sprint; however, as will be discussed in 7.1.4, this did not happen.

Relating to the Gantt chart tasks, Table 5 shows the tasks from the first sprint assigned to each team member at the start of the project.

Table 5: Initial division of tasks

<i>Team Member</i>	<i>Tasks</i>
<i>Eamonn</i>	Project manager, signal generator (Android), Android testing
<i>Gavin</i>	Oscilloscope (Android)
<i>Leonardo</i>	User interface lead for application
<i>Foivos</i>	Signal generator (iOS), iOS testing
<i>Adel</i>	Oscilloscope (iOS)

### 7.1.3 Actual Execution (Eamonn)

The project mostly adhered to the sprint plans each team member completing one task at a time (allowing for five concurrent tasks at once). The actual assignments completed in the Gantt chart can be seen in Figure 78.

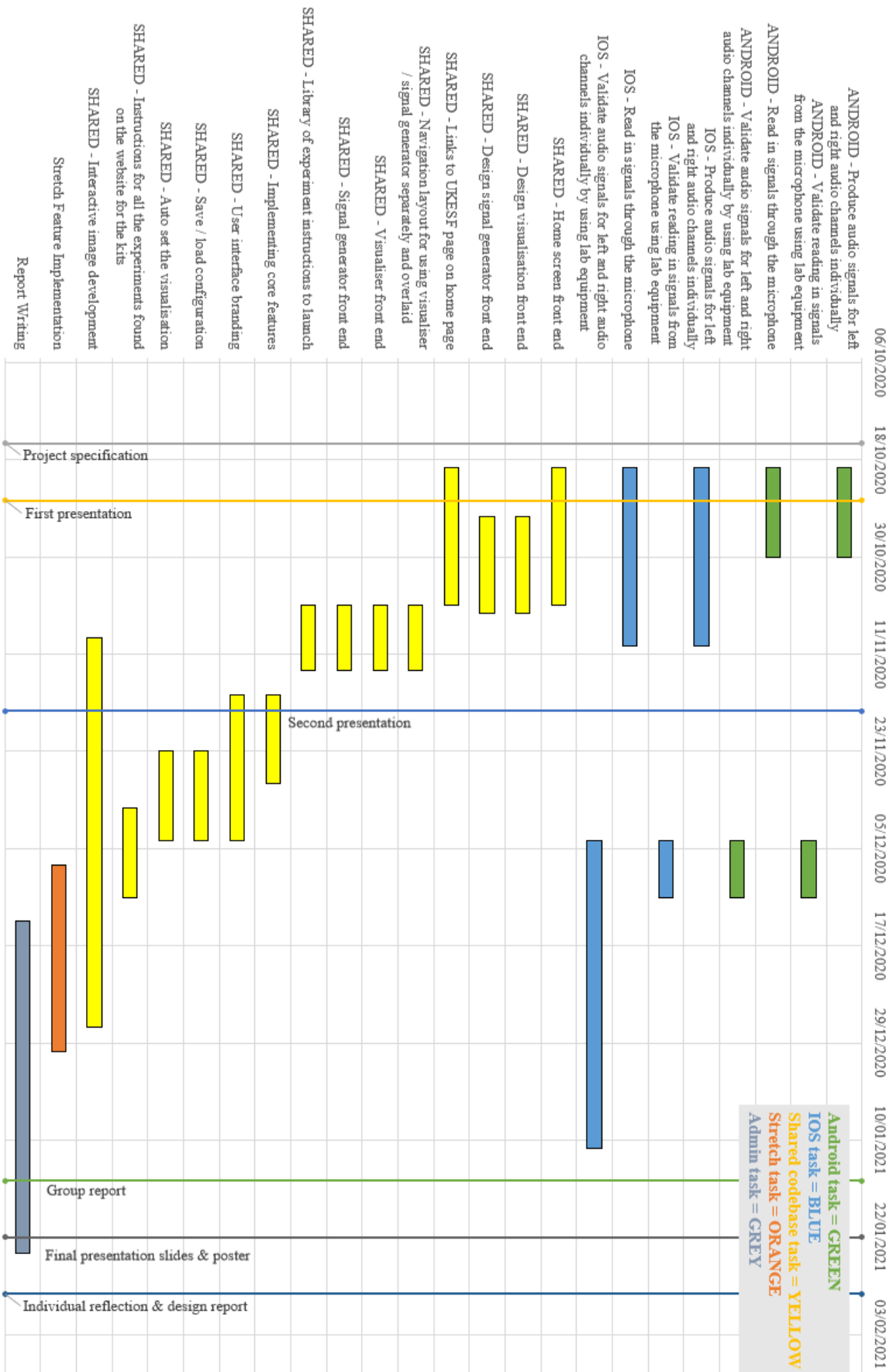


Figure 78: Actual Gantt chart execution

One of the most significant changes that required moving the most was moving the testing from the start of the project to the end. This was because some of the native implementations planned for the beginning of the project continued to prove challenging to implement completely without bugs. Since testing results would be used to report and conclude how well the application operates, this task was postponed until the native implementations had been thoroughly reviewed to remove as many bugs as possible.

Another task which overran was the interactive image for each board. This was due to both complexities of the undertaking and a lack of communication between some team members. Once the task requirement had been reiterated, the responsible team member could complete the implementation to a substantial standard in the time frame required. Internal communication is the area in which the team struggled with the most, as many project decisions were decided in the WOR, not any other times.

Another common issue was that the team members would sometimes miss meetings or turn up late to meetings despite not giving a reason beforehand. This caused some of the meetings to be delayed and overrunning. It also meant that another meeting would have to be scheduled to ensure that all the group members opinions were being considered in the decisions of the project. Luckily, the feature to record meetings was commonly used, meaning that the late or missing members could catch up on what was missed to help improve the speed of recovering the time lost. The total number of recorded times team members were late or missed meetings reached 23.

Despite these two issues in the team dynamic, the team still managed this time well. The Gantt chart planning was adhered to even whilst handling additional documentation and other requirements for other modules (or administrative tasks such as presentations or meetings) taking the focus for specific weeks.

#### **7.1.4 Actual Division of Roles** (Eamonn)

Much like the Gantt chart, the division of roles was not allocated quite precisely as they had initially been planned. With the project moving forward and there were some issues with some of the tasks. This was due to some team members struggling where others did not, and the best course of action was to reassign the team members to tasks that might have better suited their skill set. During the first sprint of the project, there were some issues in developing iOS microphone detection. Therefore, the task was reallocated from Adel to Foivos, who had some coding experience for iOS through the signal generator implementation. Therefore, Adel was tasked with producing some designs and ideas for the practical page implementation. However, the final design was heavily impacted by the inputs from the rest of the group, predominantly from Gavin and Eamonn.

The second sprint began with the team discussing the user interface of the practical page and how to align the other areas of the application. This process happened quickly and allowed the team to continue with implementation ahead of schedule. However, the iOS implementation was slightly behind, as the troubles stated above delayed the execution. As a result, Foivos continued on the back end of the iOS, with the other members moving forwards onto the front end. Leonardo continued to work on the application structure, the exercise pages and implementing changes based on feedback from the rest of the team. Gavin worked with Eamonn on implementing the user interface for the practical page, splitting the tasks for the user interaction and the graphing respectively. Adel started work

on the interactive image page due to the positively stated creativity written on the strengths and weaknesses.

For the final sprint, these tasks mostly stayed the same. However, each section of the application went through an iterative transformation, as each team member focused on getting the application themed using the UKESF colour scheme and branding. Foivos, who had completed the back-end integration, was able to work on the backend for the save/load slots, as well as the user interface for this feature.

Regarding who was responsible for each technical task, each team member wrote about in this document relate directly to their technical responsibilities. For instance, Foivos worked on the iOS backend of the oscilloscope and signal generator, and the save/load implementation. Eamonn created the backend for Android signal generation and the Music Mixer practical page graphing capabilities, alongside managing and organising the team. Gavin provided the GUI for the Music Mixer practical page and the Android oscilloscope backend. Leonardo was responsible for the general theming for the project, including the home page, about page and exercises. Adel worked on the interactive image for both the Music Mixer and the Logic and Arithmetic boards.

Beyond the final technical implementation, the team also worked on additional documentation, mostly managed by Eamonn and Gavin, who worked on the handover documentation. Eamonn and Foivos worked on the hardware testing, which provided crucial feedback for this handover documentation. Leonardo offered useful insight into the user interface side of the application and the direction it should be taken after the handover of the project. Adel continued to work on the implementation of the interactive image, overrunning the third sprint.

Table 6: Division of tasks relating to the group presentation

<i>Responsible Team Member</i>	<i>Slide Number and Title</i>
<i>Eamonn</i>	1. Introduction Slide 4. Project Planning 5. Specification 6. Project Plan 7. Planned Gantt Chart 8. Actual Gantt Chart 15. Dynamic Graphing 21. Thanks and Questions
<i>Gavin</i>	12. Native Implementations (shared with Foivos) 14. practical page 16. Application Theming and UI Design 17. Continuous Testing
<i>Leonardo</i>	10. Model View Controller 11. Current Implementations 13. Main Page and Lab Instructions
<i>Foivos</i>	2. Project Overview 3. Use Cases 12. Native Implementations (shared with Gavin) 20. Summary
<i>Adel</i>	9. Tool Overview 18. Interactive Image 19. Future Work

The most recent task was to complete the academic hand-ins for the end of the project. Three main hand-ins need to be completed: This document, a group presentation and a group poster. Before each section written in this report, the title includes the name of the author in brackets. Table 6 represents the division of tasks relating to the group presentation and who is responsible for presenting the slides. The group poster was created and edited by Gavin and Eamonn. Eamonn handled the final edit of the hand-ins to provide a consistent tone.

## **7.2 Ethical Approval and Planning of Questionnaires** (Eamonn)

### **7.2.1 Initial Rationale and Planning** (Eamonn)

A suggestion to engage with A-level physics teachers was given at the beginning of the project. Feeling it might add some useful insight into the direction of the project, and perhaps reveal some thinking about the project that the team perhaps did not consider, the decision was taken to proceed with questionnaires.

Three questionnaires were planned. The initial one was to gauge the current impact of not having enough equipment in A-level physics classes and the effect on students. It also allowed teachers to input what they might find useful in terms of a companion application or changes to the hardware.

The second questionnaire involved consideration of the design of the application at the mid-way point in the project. This was really to understand if by looking at a silent demonstration video of the layout of the application, teachers could grasp how it would be used, as the user interfaces for the signal generator or oscilloscope might be a little confusing.

The final questionnaire was much like the second questionnaire. However, it involved a silent walkthrough of the entire application and requested teachers to review if the updated design was still understandable or required changes. It was also aimed at understanding if teachers would be willing to use the application as it were in a classroom.

An ethical approval form was created through the University of Southampton's ERGO board to ensure that these questionnaires could be used within the context of this project. This took a few iterations and meant that the questionnaires were created at the start of the project. Some of the latter questionnaires had quite vague questions to capture the general feeling of the application since the specifics were not known at the time. As the questionnaires are anonymous, and the information collected within them do not relate to any personal details, the results can be kept within the project archive. The full ethical approval forms can be found in the project archive.

### **7.2.2 Evaluation of Usefulness** (Eamonn)

These questionnaires ended up being incredibly useful, but mainly for retroactive confirmation that the decisions made were in line with the opinions of A-level physics teachers (one of the end-users of the application). If, however, the project was to be repeated, these questionnaires would be done slightly different.

Firstly, the content within the questionnaires would have been more specific, with a keen interest in the advice of the direction the project could be heading in, rather than a retroactive look at what has been made. This would have involved requiring a team

member to handle updating the ethical approval through amendments and extensions to include these new questionnaires. Due to the quality of the feedback to inform the progress of the project, this would be worth the time that would be taken away from the technical side of the project.

Secondly, these questionnaires would have also aimed to retrieve more open feedback about suggestions for changes, rather than a simple evaluation of whether the application was good or bad. In the case of this project, it was helpful to know that the project was on track to please the end-users retroactively. However, there was not a high level of engagement. Perhaps knowing that there was no input in the long term success of this application, the teachers asked may not have put as much time or effort into the questionnaires as they might have if they had known that their responses had a substantial impact on the direction of the project.

## 7.3 Planning and Tools (Eamonn)

### 7.3.1 Version Control System (Eamonn)

The Version Control System used was GitHub, but more specifically, the University of Southampton GitLab. This was extremely useful to ensure that there were no possibilities of scope issues for members outside of the university from seeing the project and providing no intellectual property issues.

It was also advantageous to retrospectively see the progress on the project and the work put into each development area. Figure 79 is taken from a tool called GMaster [32], which allows a graphical representation of the version control, displaying the different branches and how they were merged for the project. Figure 79 is only a brief overview of the branch history without the intention of viewing any detail. A complete branch and commit history being displayed in appendix C.

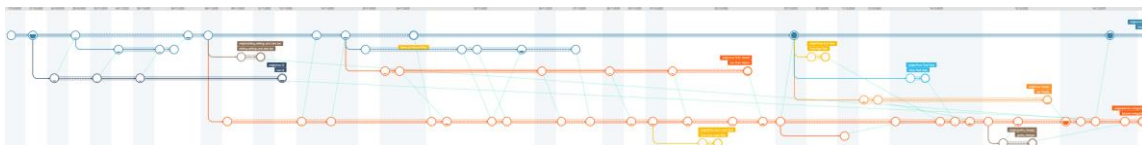


Figure 79: Overview of the GitHub branch history for the project, taken with GMaster [32]

The use of these branches meant that the team members could work on their respective features and prevent any discarding of another team members implementation if the task involved working on the same part of the code concurrently. Instead, using VCS tools to compare the differences between the merge versions and select the most relevant changes allowed the project to be combined and integrated without much hassle rapidly.

The process of using a VCS is absolutely one that would be repeated if the project were repeated. Combining a useful tool for reflecting on work done and smoothly integrating the implementations from different members with minimal effort is worthwhile. However, the difference would be using a Continuous Integration (CI) tool to allow the VCS tool to automatically generate application files on the push of each version. This would have qualified to potentially get the application rolled out to application stores for user feedback; however, implementing this in the short time given might have been slightly out of scope.



## 7.4 Risk Management (Eamonn)

### 7.4.1 Initial Risk Assessment (Eamonn)

At the start of the project, a risk assessment took place, which involved trying to plan for all of the risks that could slow the project. These potential risks were given an impact rating (a scale between 0 to 10) and a probability (a scale between 0 to 1). Together, these could be used to estimate the risk rating. Each risk was given a response that would mitigate the risk and avoid the potential damage it would cause the project.

An example of one of the risks considered was the impact of Covid-19 on the project. Figure 80 shows how this had a medium impact and a reasonably high probability, and as a result, the actual risk was relatively high. The mitigating task would be the action to take should the risk occur. A full list of the risks can be seen in appendix D.

#### Pre-Project Risk Assessment

**Imp.:** A score out of ten for how detrimental the risk would impact the project

**Prob.:** The estimated probability of the risk occurring

**Risk:** The calculated risk that this item would have on the project

ID	Risk	Hazard	Imp.	Prob.	Risk	Mitigate By
E4	There is a second UK lockdown due to Covid-19 during the project	Cannot meet in person to discuss topics. Increased stress staying in lockdown	5	0.7	3.5	Use Microsoft Teams, Trello, Facebook Messenger and any other forms of online contact necessary

Figure 80: Example of how the risk assessment was created, using lockdown due to Covid-19 as an example

### 7.4.2 Actual Risks Mitigated (Eamonn)

During the project, only a few of these risks occurred that might have impacted the project. The risks that occurred can be seen in Table 7.

Table 7: The risks that occurred during the project

ID	Risk	Hazard
P1	One of the group members travels to a different country	Impossible for all member to meet face-to-face making it harder to discuss topics
P2	One of the group members goes out of Southampton	Unlikely for all members to meet up face-to-face making it harder to discuss topics
P3	Work from other modules is hard to manage	Harder to complete tasks assigned with much work from other modules
E2	Ethics application takes longer than expected to be accepted.	Feedback from teachers is received late resulting in possible project delays
E3	Cannot make a booking to meet up in person	Cannot discuss topics in person may make it harder to understand information
E4	There is a second UK lockdown due to Covid-19 during the project	Cannot meet in person to discuss topics. Increased stress staying in lockdown
E5	Cannot meet up face to face with the client	Harder to understand what the client wants

P1, P2, E3, and E5 were caused because of the same factors; therefore, the solution was the same for all three. Meetings from the start of the project were planned online, though Microsoft Teams and little work needed to be carried out in person. It also helped that a couple of team members bubble together, which meant that the small in-person work could be done in a team. The laboratory tasks were also carried out as usual, as the hardware supplied was enough to split between the team to test independently.

P3 only occurred towards the end of the project, since this is when most of the other courses have their deadlines for coursework. The mitigating factor involved managing the team well by ensuring good communication of when deadlines were. This allowed informed decisions to be made about the amount of work to be undertaken from week to week.

E2 impacted the project near the beginning. The initial survey needed to be sent as soon as possible since it was the basis for many of the decisions for the project. The ethical approval process took longer than expected and caused this information to be returned later than expected. Luckily, the data was just as useful in hindsight as it would have been before starting the project. The initial development included only the hardware interfacing, so the feedback could still be used in to design the user interface features.

E4 occurred for four weeks between the 4<sup>th</sup> November until the 5<sup>th</sup> of December. Luckily, due to the mitigating circumstances from P1, P2, E3, and E5, much of the communication remained unchanged during this period. The team members in a bubble were also still allowed to continue to do so during the lockdown period. The uncertainty of the university laboratories closing prompted by this risk was a useful motivator to complete testing early.

## 7.5 Strengths and Weaknesses Matrix (Eamonn)

### 7.5.1 Initial Strengths and Weaknesses (Eamonn)

Skill	Team Members & Skill Rating /10					Team Average	Team Variance	Top Score	Top People
	Eamonn	Gavin	Adel	Leo	Foivos				
Leadership	10	9	7	5	7	7.60	5	10	Eamonn
General Management	9	9	5	8	6	7.40	4	9	Eamonn, Gavin
Time Management	8	7	5	7	8	7.00	3	8	Eamonn, Foivos
Research	5	3	5	5	5	4.60	2	5	Eamonn, Adel, Leo, Foivos
Creativity	7	3	10	8	7	7.00	7	10	Adel
Report Writing	7	6	4	7	8	6.40	4	8	Foivos
Report Formatting	8	5	4	7	9	6.60	5	9	Foivos
Conscientious	7	9	8	8	8	8.00	2	9	Gavin
Social Awareness	8	9	7	7	8	7.80	2	9	Gavin
General Analysis	7	7	8	6	7	7.00	2	8	Adel
Data Analysis	5	6	8	5	7	6.20	3	8	Adel
Strategical	8	6	10	6	7	7.40	4	10	Adel
Legal Knowledge	3	3	3	7	3	3.80	4	7	Leo
Reasoning	7	7	8	7	7	7.20	1	8	Adel
Teamwork	8	6	7	7	6	6.80	2	8	Eamonn
Adaptability	8	5	7	8	7	7.00	3	8	Eamonn, Leo
Practical	7	8	5	6	6	6.40	3	8	Gavin
Resourceful	6	6	8	5	5	6.00	3	8	Adel
Sociable	7	9	7	7	6	7.20	3	9	Gavin
Inventive	7	4	8	5	7	6.20	4	8	Adel
C / C++ Coding	6	6	8	3	8	6.20	5	8	Adel, Foivos
Javascript Coding	8	2	2	6	8	5.20	6	8	Eamonn, Foivos
Java Coding	9	2	2	9	2	4.80	7	9	Eamonn, Leo
Hardware Design	6	8	6	1	1	4.40	7	8	Gavin
Hardware Interfacing	5	8	5	1	6	5.00	7	8	Gavin
App Design	7	6	7	8	7	7.00	2	8	Leo
GUI Design	8	8	6	9	9	8.00	3	9	Leo, Foivos
Dart Coding	0	0	0	0	0	0.00	0	0	
Swift Coding	4	2	2	4	3	3.00	2	4	Eamonn, Leo
Individual Average	7.074074	5.827586	5.931034	5.931034	6.481481	6.68	3.30	8.35	
	Eamonn	Gavin	Adel	Leo	Fivos	Average	Average	Average	

Figure 81: Initial strengths and weakness matrix

At the start of the project, to ensure that the team could assign tasks relating to their strengths and weakness, a complete matrix of the team members' capabilities was created. The full matrix of the responses given can be seen in Figure 81. Note: Dart Coding has been filled out with 0 as the team had yet to choose Flutter as their tool.

The most notable aspect of the group strengths and weaknesses includes a team average of 6.68, indicating an overall strongly performing group. The team had a general lack of strengths in research, legal knowledge, and Swift coding.

### 7.5.2 Strengths and Weaknesses After Project (Eamonn)

Throughout the project, the team members were able to work on different aspects of their strengths and weaknesses. The team filled out the strengths and weaknesses matrix at the end of the project, without reference to the original. These responses can be seen in Figure 82.

The differences between this matrix and the initial one show that the team have increased their average from 6.68 to 6.88. This shows how each team member has improved their skillset throughout the project. The most remarkable individual improvement was Gavin, who was able to increase an average of 1.17 points. The team also seemed to become more aligned with their skills, noting that the team variance fell from 3.3 to 2.5. Unfortunately,

the top scope average of the team also fell from 8.35 to 8.20. This is still remarkably high and shows just how strongly the team performed throughout the project, with the initial strengths chart already showing a strong team to begin, improving on that shows the commitment and dedication of the team to the project.

Skill	Team Members & Skill Rating /10					Team Average	Team Variance	Top Score	Top People
	Eamonn	Gavin	Adel	Leo	Foivos				
Leadership	10	9	6	6	6	7.40	4	10	Eamonn
General Management	9	9	6	8	8	8.00	3	9	Eamonn, Gavin
Time Management	9	8	6	8	7	7.60	3	9	Eamonn
Research	7	6	7	7	6	6.60	1	7	Eamonn, Adel, Leo
Creativity	5	5	7	8	6	6.20	3	8	Leo
Report Writing	7	6	4	7	8	6.40	4	8	Foivos
Report Formatting	8	6	4	7	8	6.60	4	8	Eamonn, Foivos
Conscientious	7	9	6	8	6	7.20	3	9	Gavin
Social Awareness	7	9	7	7	7	7.40	2	9	Gavin
General Analysis	7	7	7	7	8	7.20	1	8	Foivos
Data Analysis	6	6	7	5	8	6.40	3	8	Foivos
Strategical	8	9	8	7	7	7.80	2	9	Gavin
Legal Knowledge	3	3	5	5	3	3.80	2	5	Adel, Leo
Reasoning	8	8	8	8	7	7.80	1	8	Eamonn, Gavin, Adel, Leo
Teamwork	8	7	6	7	6	6.80	2	8	Eamonn
Adaptability	8	7	7	8	8	7.60	1	8	Eamonn, Leo, Foivos
Practical	7	9	6	5	9	7.20	4	9	Gavin, Foivos
Resourceful	6	6	6	7	5	6.00	2	7	Leo
Sociable	6	9	6	6	6	6.60	3	9	Gavin
Inventive	7	7	7	8	6	7.00	2	8	Leo
C / C++ Coding	6	6	8	5	8	6.60	3	8	Adel, Foivos
Javascript Coding	8	3	3	7	6	5.40	5	8	Eamonn
Java Coding	9	5	3	9	3	5.80	6	9	Eamonn, Leo
Hardware Design	6	8	5	3	7	5.80	5	8	Gavin
Hardware Interfacing	8	9	5	3	9	6.80	6	9	Gavin, Foivos
App Design	8	8	7	8	8	7.80	1	8	Eamonn, Gavin, Leo, Foivos
GUI Design	8	8	7	9	8	8.00	2	9	Leo
Dart Coding	8	7	7	8	8	7.60	1	8	Eamonn, Leo, Foivos
Swift Coding	5	4	3	3	6	4.20	3	6	Foivos
Individual Average	7.206897	7	6	6.689655	6.814815	6.88	2.50	8.20	
	Eamonn	Gavin	Adel	Leo	Fivos	Average	Average	Average	

Figure 82: Final strengths and weaknesses matrix

## 7.6 Achievements and Results (Eamonn)

Considering the planning at the start of the project, with the Gantt chart, the choice of tools and the specification decided compared to the actual Gantt chart, the effectiveness of these tools, and the completion of all core specifications have managed an outstanding achievement.

The ability to implement the hardware interfacing on both iOS and Android applications, whilst continuing to consider the user interface, through branding and end-user experience viewpoints has allowed the project to produce an application has a few bugs and looks professional.

On top of the core technical features and implementations, the team also produced supporting documentation in professional presentations, videos, images and further plans. These additional documentations were created to provide the team who will be continuing work on the implementation to understand the teams' vision with the application. They also aim to guide the takeover team in research areas that had already been done to avoid lost time in retracing the steps of the original group.

In addition to all of this, Stewart Edmondson of the UKESF (the client of the project), gave feedback on the project. In terms of the approach, he stated, “the approach showed good [innovation], the execution was done really well, and the design solution was cleverly thought through.” Relating to technical work Stewart was “impressed by the professional approach”, stating that he “was very pleased with the outcome, which showed that the group understood both the context and the design brief.” In terms of communication and negotiation, Stew thought the group “provided excellent weekly updates” with “overall, communication [being] a strength.” Stewart thought “a lot has been achieved in a short space of time” being “particularly pleased with the Group’s (positive) response to the comments and feedback on their initial design brief”, stating “it was really good to see our comments being acknowledged and acted upon.” Commenting on the group, Stewart thought “the group were really well-motivated and very organised; they seemed to work well together.” Stewart's thoughts are that the project is “worthy of being used” as it has “met all major goals”. The originally formatted feedback can be seen in appendix E.



## 8 Conclusion

### 8.1 Specification (Eamonn)

According to the initial specification discussed in 3, the following points have been complete:

- *Visualisation of an input signal from the microphone*
  - *44.1kHz sample rate at 16-bit resolution*
  - *Ability to change the scale, relative phase to the generated signals and auto set*

With the Music Mixer practical page operating using a 44.1 kHz 16-bit PCM for iOS and Android, with the ability to trigger and have the signal generator graphs change scale based on the signal generated, all these points have been completed.

- *Signal generator through the audio port*
  - *A maximum of 44.1kHz maximum voltages is 2V peak-to-peak at a 16-bit resolution at loads of 100-600 Ohms impedance*
  - *Two individual mono-channel signals on the left and right audio channel*
  - *Sine, square, triangle waves, music samples as default options*
  - *Ability to change the amplitude, relative phase, frequency*
  - *Can be individually turned off and on (if both are off, then music can be played)*
  - *To be directly visualisable alongside the input signal from the microphone*

With the Music Mixer practical page able to produce two independent signals that can play sine square or triangle waves with relative phase differences, between 20-5000 Hz at different amplitudes the top four points have been satisfied. The practical page allows the signal generator graphs to be displayed alongside the microphone input, and if the signals are off, music can be played out of the auxiliary port.

- *Instruction sets for experiments*
  - *Targeted at students to follow (enabling simple lesson plans for teachers)*
  - *Covers all the material already available on the webpages*
  - *Provides contextual background knowledge and theory about the experiment*

With instructions taken from the web page for both the Music Mixer and Logic and Arithmetic kits each with an introduction page explaining the purpose of the exercise, these points have been satisfied.

- *User interface*
  - *Targeted primarily at students to use, but also with teachers in mind*
  - *Links to UKESF & University of Southampton tools*
  - *Branding and colour scheme matching UKESF and Electronics Everywhere*
  - *Focusing on functional features to support experiments, without too many additional to avoid an overwhelming interface*
  - *Save/load slots to retain specific configurations*

The theming of the application has focused on the UKESF colour scheme, with links to additional information for both UKESF and the University of Southampton. The practical page is simple enough for students to understand (demonstrated in the survey responses) and allows the user to save and load configurations for different exercises.

There were some of the stretch goals implemented into the application; these included the following:

- *User interface*
  - *Pinch zooms and edits for signals*
  - *Interactive pictures of boards to give information about the operation*
  - *Addition instructional tasks to produce certain output signals (within tolerances) based on changing the circuit/signal configuration (providing the more adept students with a chance to explore the more difficult application of physics, i.e. using these two output signals, and using the board, produce a signal that is a mix of the first two)*

The pinch zooms were not implemented. However, the ability to edit signals accurately was added with a text input. The interactive pictures of the boards with additional information on background theory and operation. The instructions included in the application have a step-by-step format; however, do not check for any output signals.

- *Logic analyser*
  - *Information on the board included within the application*
  - *Instructions for the experiments on the logic analyser found on the website*
  - *Produce a technical plan for implementing a logic analyser with the application (including hardware if required)*

The first two points have been completed with the inclusion of the interactive image and the Logic and Arithmetic kit exercises. The technical plan for implementing a logic analyser with the application was discussed in the handover documentation.

## 8.2 Concluding Discussion (Foivos)

This project has successfully demonstrated the design, implementation and testing of a cross-platform signal generator and oscilloscope application referred to as “Electronics Everywhere Application”. Following the specification referenced in 3, the project has successfully produced all the core features and some stretch goals. Using Flutter, a software development library created by Google, combined with native iOS and Android code, the signal generator and oscilloscope functionality were implemented.

This application has a target audience of A-level physics and computer science students, along with teachers. The primary use case of the application is to be used alongside the already developed UKESF Electronics Everywhere kits. The application included branding and colour schemes from the UKESF; something the client stressed was a crucial part of the project.

The supporting content surrounding the kits, such as the experiments found online, was added to the application as interactive instructions. To support the exploration of the kits and the contributing bodies, all relevant Electronics Everywhere and University of Southampton links are referenced in the application. Each kit also features an interactive image, allowing students to familiarize themselves with the boards and learn about the electronic concepts included on them.

The client requested an application that was more polished rather than filled with buggy features at the end of development. The team ensured that the core features were implemented without bugs. Some stretch features that could not be implemented in a bug-free state within the time frame are explored and discussed in 9.



Some handover resources were created that includes not only further work but also promotional materials, tutorials, walkthroughs, edited content, and results from the surveys and testing. This documentation is crucial to the team continuing the project to understand the work already done and avoid repeating the same work.

The applications performance was tested at the end of the project. The results found that on both Android and iOS, the margin of error was on average below 1%. The testing justified the decision not to include an amplitude measurement on the y-axis of the practical page, as there is no way to determine the amplitude of the output values. Similarly, the input amplitude can also vary depending on the hardware connected. These tests proved the application was ready for use, with the additional requirement of an in-line attenuator.

Throughout the project, A-level teachers helped to provide feedback on the justification for the project and input crucial feedback on the design process. The general feedback involved discussion that A-level classes had limited access to oscilloscope and signal generators, so an application that could emulate these features would be beneficial. The intermediate solution design had positive feedback, stating that 93% of teachers either agreed or strongly agreed that students would use the application. The final survey only received three total responses, and so the results are not included as they would not provide a genuine discussion of the application.

The client has stated they were impressed with the project overall, both in the approach and the achievements. The project was “very organised” with “communication [being] a strength”. The group “understood both the context and the design brief” in the view of the client, with the approach showing “good [innovation]”. The client believes the application is “worthy of being used” having met “all major goals”.

The project was managed successfully, with potential risks mapped out in a risk matrix and assigned roles according to individual strengths and weaknesses. The initial Gantt chart was mostly adhered to, with only a couple of tasks overrunning or being rescheduled to accommodate the new timeline. With productive Weekly Operation Review meetings, the team completed the specification on time and professionally, even managing to hand over additional documentation.

This report was written with Microsoft Word, reporting a word count of 24696 (starting from the title of 1, until the final word before 10).



## 9 Further Work (Gavin)

Some additional features were investigated and discussed during the project but were not implemented due to timing constraints. This section discusses these features with initial designs, considerations, and descriptions of potential implementations.

### 9.1 practical page Tutorial (Gavin)

There are instructions, both written and in video format demonstrating how to use the application, but this is not currently embedded within the application. The practical page is the most complex, and feature-full section, so might benefit from an embedded tutorial. The instructions could be tooltips that show the functionality on-screen. An example of how this might look is shown in Figure 83.

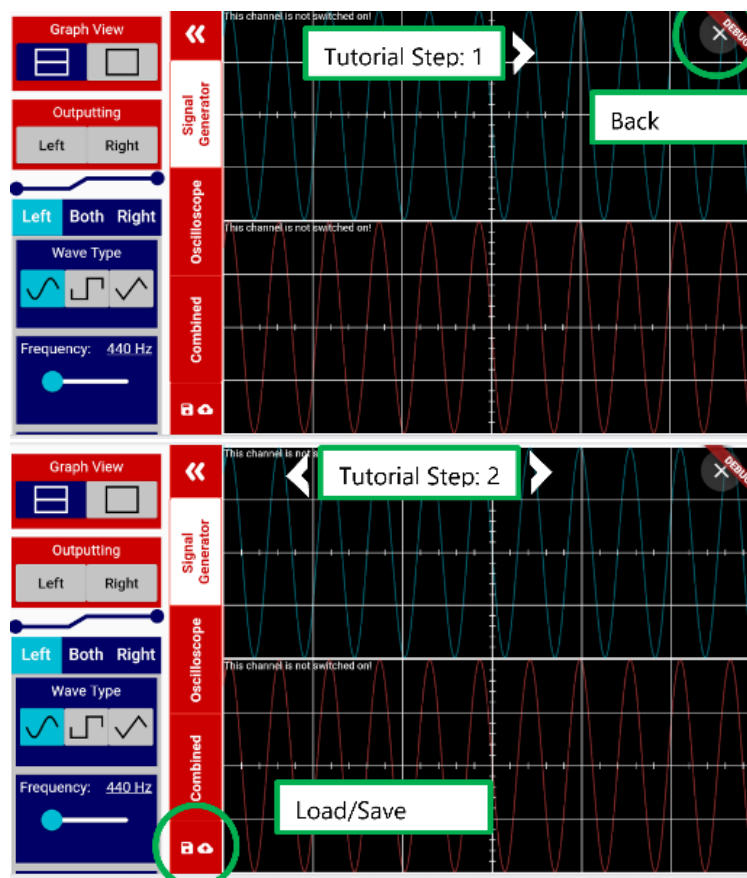


Figure 83: Mock-up design of practical page tutorial

The tutorial should only appear once, on the first launch of the practical page. If the user would like to repeat the tutorial, there should be a method to do this. These tooltips could end with an embedded video of the application walkthrough discussed in 6.2.2.

Tooltips could be used to display information similar to Figure 83. Using the `SharedPreferences` library would allow the application to display the tooltips on the first time the practical page is opened, setting a boolean variable after completion. A sequence of instructions could lead the user through some typical use cases demonstrating the functionality.

## 9.2 Links from the Experiment Instructions to the practical page (Gavin)

Currently, when a user is following the instructions from the exercises embedded in the application, to get to the practical page, they must leave the instructions, navigate the Music Mixer information screen to reach the practical page. A more intuitive implementation should directly navigate the user from the instruction page to the practical page. A diagram explaining this user experience can be seen in Figure 84. This can streamline the process of following instructions by automatically setting the required parameters for the step of the exercise on the practical page. For example, if the experiment requires a 440 Hz sine wave, the practical page has this pre-loaded when launched from the instructions page.

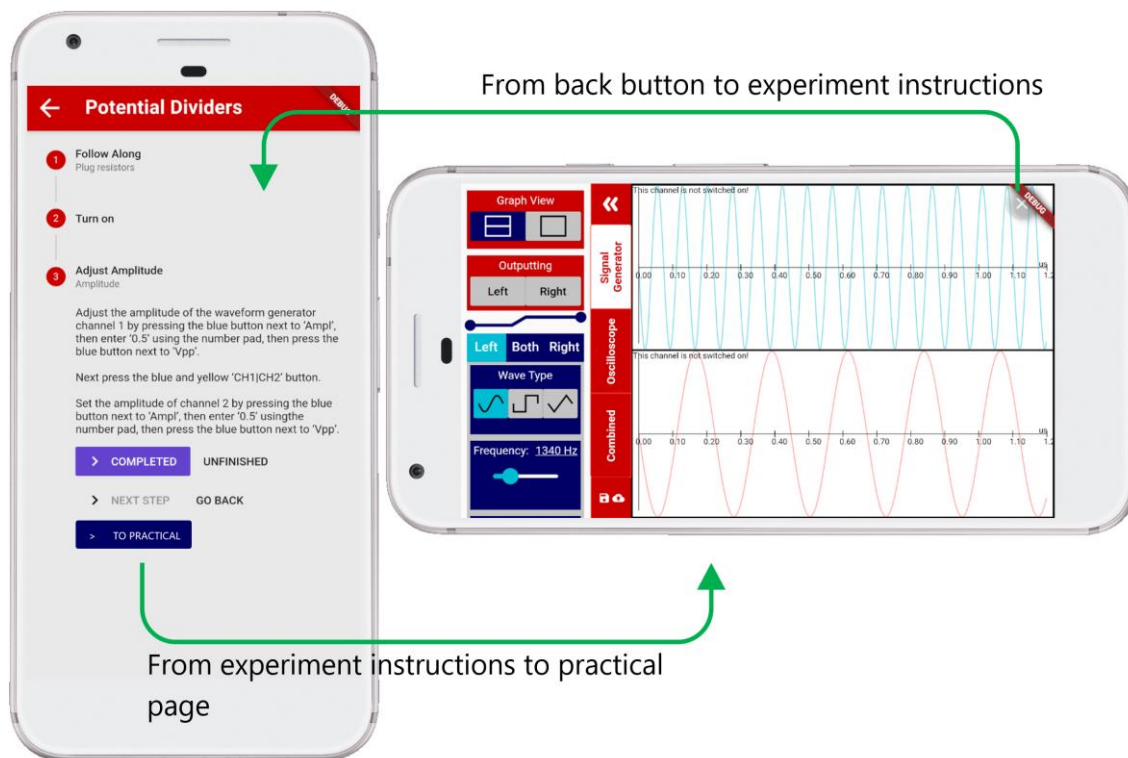


Figure 84: Mock-up diagram of the transitions to and from the instructions and practical page

The button to launch into the practical page would primarily affect users running experiments and reading instructions with a single phone. If multiple users each have the application on their devices, one device could have the practical page open, and the other has the instructions open, removing the requirement for this implementation. After the practical page has been opened from the instructions, the implementation should be clear that closing the practical page will take the user back to the instructions. This can either be left to the intuition of the user, with the exit button or made evident through a user interface indication. Having the ability to set parameters automatically could save time for the user but might take away from the engagement and understanding.

A method of implementing this feature would be adding a button on the instructions, as shown in Figure 85. Returning to the instructions could be an extra option that pops up from the floating exit button currently implemented on the practical page.



Figure 85: Placeholder button to launch to the practical page from the instructions

### 9.3 Interactive 3D CAD Model (Gavin)

The interactive image could be further improved by replacing the 2D image with a 3D model that would allow the user to visualize all aspects of the kits using finger gestures such as “pinch to zoom” or “swipe to rotate”. The model could also zoom and move to a specific element whenever a particular section is tapped.

A simple CAD model might have less detail due to the computation required for a highly detailed model. Having the 3D model could allow users to freely explore the board in any orientation or perspective, potentially increasing engagement in exploring the board. The method for navigating the model should be intuitive so that users can quickly explore the board. The impressiveness of a 3D model compared to a 2D model could encourage users to explore the board in more depth, and spend more time discovering the features and concepts used on the boards.

A CAD model of the boards would have to be created and implemented into the application. A potential method of this would be the use of a Flutter package `model_viewer` [33]. Tooltips could appear when certain board areas are being observed, which could be selected to show further information, much like the current implementation of the interactive image.

### 9.4 Cross-Platform Support for Web Browsers (Gavin)

Flutter has support to convert the current application to work with web browsers, and this would be another good addition to help remove the barriers of use for the application.

If this application is supported in web browsers, this would be widely accessible, encouraging more widespread usage. With a web application, the hardware can potentially vary a lot more than on phones, which might cause issues when receiving and sending information via a headphone port. This could also cause issues with requesting permissions to access hardware resources. There are also legal accessibility requirements that would need to be considered.

The cross-platform on web browsers could be enabled while also ensuring enough warning about the difference in support. Most of the application should work in a similar way to the mobile application. However, any differences should be noted on the application. The benefit of using the implementation already developed is to leverage the single code base and concurrent development.

### 9.5 Logic and Arithmetic Hardware Support (Gavin)

The Logic and Arithmetic board is included in the application through the interactive board walkthrough and the ten exercise instructions from the lab notes. Further support would include a separate piece of hardware that interfaces between the board and the phone so that the application could contain a practical page, much like the Music Mixer.

The application would provide tools such as a logic analyser that could be used to visualise digital signals from the board.

The hardware required to interface with the Logic and Arithmetic board would be more extensive than interfacing with the Music Mixer board, as it would not be suitable to interface using the headphone port. The Logic and Arithmetic board would need updating to include support for interfacing with an application. The signals being sent to the application would be digital, and so an interface using USB or Bluetooth would be appropriate. Adding this hardware might be costly but could potentially produce a patent for a connector that works with both USB-C, lightning and USB-A connection to ensure that any device can use this logic analyser.

A potential implementation would include a separate piece of hardware, probably a PCB, that includes probes that can connect to the Logic and Arithmetic board. Some of the implementation would be software to interpret these signals within the application. This is estimated to communicate using USB or Bluetooth and could send multiple digital streams containing data on the logic values (or potentially other information the probe can detect). If an external tool is not required, the board could have the hardware directly embedded so that the phone can connect directly to the board.

## 9.6 Slide Out Gesture for Settings Bar (Gavin)

The settings bar is currently operated using the arrow button to expand and hide the settings bar with an animation. A potential alternative to the button-controlled behaviour is implementing a gesture to slide out the settings bar that “follows” the motion. This could either replace the current button or work in addition to it.

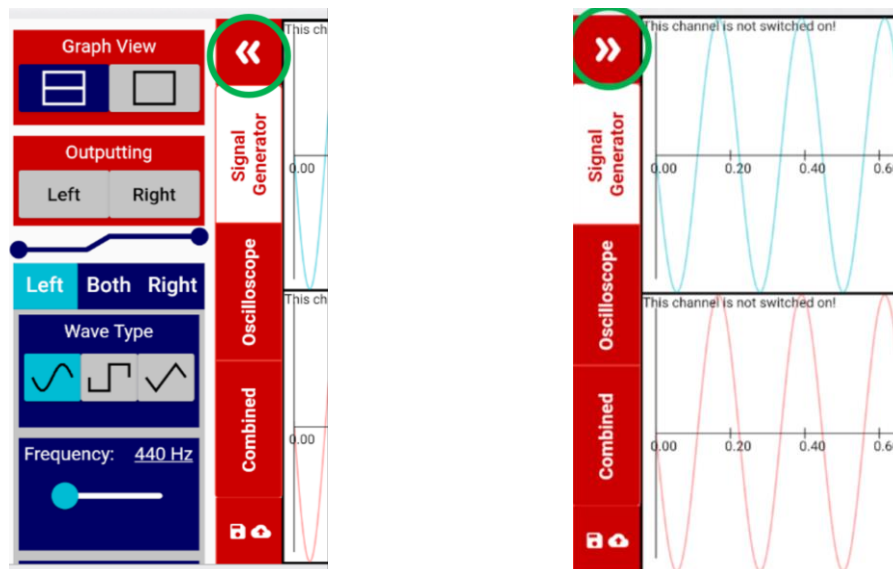


Figure 86: Practical page parameter menu, showing the collapse button (Left) expanded (Right) collapsed

Both gestures and buttons are commonly used in applications to open and close menus, similar to the parameter menu. A user will likely try one of the options and then resort to the other, however, to provide the most accessibility, a slide motion would improve the number of users who naturally discover the sidebar operation. If the button implantation were removed, the exit button for the page could replace the current button for the sliding in and out of the sidebar. Another consideration is if the button to open or close the menu

is removed, when the menu is collapsed a user might not realise that it can be expanded. If the button was removed, a different visual indicator that the sidebar can be expanded should be added. A slide-out gesture could either replace or be an addition to the current button. The exit button could be re-positioned in place of the open/close button.

## 9.7 Pinch to Zoom for the practical page (Gavin)

The current method of changing the oscilloscope scale (and all the graphs on the combined) is by moving the horizontal scale slider. A pinch gesture could be used in addition to this to change this scale.

A pinch gesture might be more intuitive than a slider to zoom into the graphs. This feature would not interfere with the current slider implementation. There is a potential for a pinch gesture could cause issues with other gestures and motions, such as a sliding gesture to open and close the parameter menu. However, this is unlikely because the slide gesture uses a single finger, with a pinch requiring two. The scale of the graphs in the signal generator mode is automatic and so cannot be manually changed. If a pinch to zoom were implemented, it would be expected to change the zoom for all graphs. This would mean adding an ability to scale manually and be overridden only when the signal frequency has been updated. The current scale only affects the horizontal axis, and it is recommended that the pinch to zoom feature aligns with this implementation.

## 9.8 Triggering Based on Movable Trigger Point (Gavin)

Triggering is currently present in this application, but the level at which the signals are triggered is 0. The ability to change the trigger level could be a useful addition and is standard in most oscilloscopes. This would allow more complicated signals to be triggered, allowing for better analysis of constructive behaviour.

The current implementation of the trigger level at 0 assumes only a single negative edge in a period. For many signals, this will not be the case. The only apprehension about adding this feature is that many A-level students will not know what a trigger level is, so this feature might not be required. The inclusion of this feature would help to generalise the oscilloscope functionality for more uses beyond the use case with the Music Mixer board. If this feature is implemented, the signal generator graphs would also need to appear “triggered” in the combined view.

A potential solution would be to include this movable trigger option but have a popup that appears the first time the user moves the slider, explaining what it does and how it works. This would allow the functionality without requiring students to have prior understanding. The signal generator display would also react to this trigger level, reducing the complexity for the end-user. The triggering for the signal generator would operate under two considerations. The first is that the user is in the combined mode (assuming the same phone is used for input and output). The second consideration is that the user has set the trigger point to one that is common for both the signal generator and the oscilloscope.

## 9.9 Support for Microphone Inputs to Phone Directly from the Music Mixer Board (Gavin)

The Music Mixer board does not currently have an output that maps to a microphone input on the phone, nor includes a large enough impedance for a mobile phone to detect it as an input without an in-line attenuator. The current solution is the inclusion of a headphone splitter that can correctly map the output of the Music Mixer board to the microphone connection. A potential solution is to add these functionalities to the output of the board so that a no in-line attenuation or splitter needs to be added since it is already on the board. The specific hardware requirements are also discussed in 5.4.2.

The audio output may be used for audio playback via headphones or speakers directly from the Music Mixer board and may require a higher drive than that for a microphone. For a microphone to be detected on phones, an impedance needs to exist between the microphone connection and GND. The additional cost on the board may or may not outweigh the cost of adding an in-line attenuator required for easy detection of the microphone on a phone. Adding a specific output for a microphone would prevent using a single phone to both output to the board and take an input. The best solution is dictated by the number of phones typically interfacing with the board. If a single phone is the most common case, an “all in one” port would be beneficial, but if three phones were the most common case, having three separate ports (similar to the current layout) would be more beneficial. Having one port on the board would mean extra adapters if multiple phones were to connect with a signal board, whereas if there were various ports on the board, more adapters would be needed for a single phone to interface with the board.

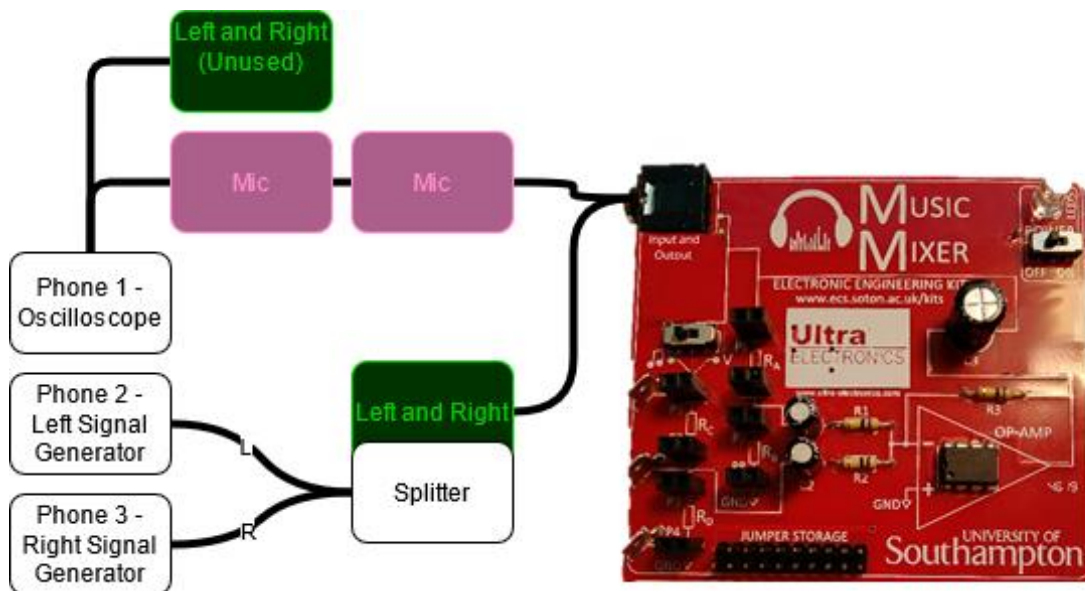


Figure 87: Use case of the board redesign with TRRS connection to three phones with splitters

One option is to replace the inputs and outputs on the board to be a single 4-Segment Plug [34] into the board. In doing so, only a single connection cable would be required from the phone to the board (a simple TRRS cable). This assumes that the microphone output conforms to the requirements stated in 5.4.2 for microphone detection. There are two benefits from this, a reduced cost in ports placed on the board, and a reduced cost in the number of adapters required to interface to a single phone. The downside would be that



splitters would no longer be as quickly connected to the board. However, it could still be done with the recommended splitters and simultaneously used with multiple phones. A mock-up is shown in Figure 87. The real benefit of this solution is shown in Figure 88, where a single phone interfaces with the board.



Figure 88: Use case of the board redesign with TRRS connection to a single phone that is used as both oscilloscope and signal generator

Another option is to add an output to the board that maps the output to the microphone section to include enough impedance for microphone detection. This port could be used to connect to a device for automatic microphone detection. The existing output port would remain unchanged and could be used to output to headphones and speakers as before.



# 10References

- [1] Flutter, “Flutter Plugins: Shared Preferences,” Flutter, [Online]. Available: [https://github.com/flutter/plugins/tree/master/packages/shared\\_preferences](https://github.com/flutter/plugins/tree/master/packages/shared_preferences). [Accessed 2020].
- [2] Flutter, “Flutter Plugins: URL Launcher,” Flutter, [Online]. Available: [https://github.com/flutter/plugins/tree/master/packages/url\\_launcher](https://github.com/flutter/plugins/tree/master/packages/url_launcher). [Accessed 2020].
- [3] Once10301, “Flutter Plugins: Permission,” [Online]. Available: <https://github.com/once10301/permission>. [Accessed 2020].
- [4] FlutterCommunity, “Flutter Plugins: Flutter Launcher Icons,” FlutterCommunity, [Online]. Available: [https://github.com/fluttercommunity/flutter\\_launcher\\_icons](https://github.com/fluttercommunity/flutter_launcher_icons). [Accessed 2020].
- [5] anarchuser, “Flutter Plugins: Mic Stream,” [Online]. Available: [https://github.com/anarchuser/mic\\_stream](https://github.com/anarchuser/mic_stream). [Accessed 2020].
- [6] UKESF, “About,” UKESF, [Online]. Available: <https://www.ukesf.org/about/>. [Accessed 2020].
- [7] UKESF, “UKESF Infographic,” UKESF, [Online]. Available: <https://www.ukesf.org/wp-content/uploads/2018/03/17-UKESF011-UKESF-Infographics-Update-A4-1.pdf>. [Accessed 2020].
- [8] UKESF, “Electronics Everywhere,” UKESF, [Online]. Available: <https://www.ukesf.org/employers/electronics-everywhere/>. [Accessed 2020].
- [9] University of Southampton ECS, “Electronic Engineering Kits,” University of Southampton, [Online]. Available: <https://www.ecs.soton.ac.uk/outreach/kits>. [Accessed 2020].
- [10] University of Southampton ECS, “A-Level Physics Music Mixer Kit,” University of Southampton, [Online]. Available: <https://www.ecs.soton.ac.uk/outreach/kits/physics-music-mixer-kit>. [Accessed 2020].
- [11] University of Southampton ECS, “A-Level Computer Science Logic and Arithmetic Kit,” University of Southampton, [Online]. Available: <https://www.ecs.soton.ac.uk/outreach/kits/computer-science-logic-and-arithmetic-kit>. [Accessed 2020].
- [12] University of Southampton ECS, “A-Level Computer Science: Logic and Arithmetic Kit Training Handbook,” [Online]. Available: [https://www.ecs.soton.ac.uk/sites/www.ecs.soton.ac.uk/files/alevel-cs-training-lab-notes\\_0.pdf](https://www.ecs.soton.ac.uk/sites/www.ecs.soton.ac.uk/files/alevel-cs-training-lab-notes_0.pdf). [Accessed 2020].
- [13] R. Silva, “PCM Audio in Stereo and Home Theatre,” Lifewire, 25 11 2019. [Online]. Available: <https://www.lifewire.com/what-is-pcm-1846928>. [Accessed 2020].
- [14] C. E. Shannon, “Communication in the Presence of Noise,” *Proceedings of the IRE*, vol. 37, no. 1, pp. 10-21, 1949.
- [15] Elprocus, “Pulse Code Modulation and Demodulation,” Elprocus, [Online]. Available: <https://www.elprocus.com/pulse-code-modulation-and-demodulation/>. [Accessed 2020].
- [16] Android, “AudioTrack,” Android, [Online]. Available: <https://developer.android.com/reference/android/media/AudioTrack>. [Accessed 2020].
- [17] Android, “AudioRecord,” Android, [Online]. Available: <https://developer.android.com/reference/android/media/AudioRecord>. [Accessed 2020].

- [18] Apple, “Audio Unit Hosting Fundamentals,” [Online]. Available: [https://developer.apple.com/library/archive/documentation/MusicAudio/Conceptual/AudioUnitHostingGuide\\_iOS/AudioUnitHostingFundamentals/AudioUnitHostingFundamentals.html](https://developer.apple.com/library/archive/documentation/MusicAudio/Conceptual/AudioUnitHostingGuide_iOS/AudioUnitHostingFundamentals/AudioUnitHostingFundamentals.html).
- [19] Statcounter, “Statcounter,” [Online]. Available: <https://gs.statcounter.com/os-market-share/mobile/united-kingdom>. [Accessed 2020].
- [20] Google, “Explore search interest for Xamarin, React Native, Flutter, Phonegap, Ionic by time, location and popularity on Google Trends,” Google, [Online]. Available: <https://bit.ly/3siO8vk>. [Accessed 2020].
- [21] Keuwlsoft, “Function Generator,” Keuwlsoft, [Online]. Available: <https://play.google.com/store/apps/details?id=com.keuwl.functiongenerator&hl=en&gl=US>. [Accessed 2020].
- [22] P. Reinholdtsen, Vaclav and W. Schweizer, “AUDio MEasurement System,” SourceForge, [Online]. Available: <https://sourceforge.net/projects/audmes/>. [Accessed 2020].
- [23] G. E. Krasner and S. T. Pope, “A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 System,” ParcPlace Systems, California, 1988.
- [24] Atlassian, “What is version control,” Atlassian, [Online]. Available: <https://www.atlassian.com/git/tutorials/what-is-version-control>. [Accessed 2020].
- [25] M. Gallagher, “An iOS tone generator (an introduction to AudioUnits),” 20 October 2010. [Online]. Available: <https://www.cocoawithlove.com/2010/10/ios-tone-generator-introduction-to.html>.
- [26] Android, “Android Developers - AudioRecord,” [Online]. Available: <https://developer.android.com/reference/android/media/AudioRecord>. [Accessed 2020].
- [27] UKESF, “UKESF Homepage,” UKESF, [Online]. Available: <https://www.ukesf.org/>. [Accessed 2020].
- [28] U. o. Southampton, “Training Handbook,” University of Southampton, [Online]. Available: [https://www.ecs.soton.ac.uk/sites/www.ecs.soton.ac.uk/files/Training%20Handbook\\_0.pdf](https://www.ecs.soton.ac.uk/sites/www.ecs.soton.ac.uk/files/Training%20Handbook_0.pdf). [Accessed 2020].
- [29] U. o. Southampton, “Logic and Arithmetic Electronic Engineering Kit,” University of Southampton, [Online]. Available: [https://www.ecs.soton.ac.uk/sites/www.ecs.soton.ac.uk/files/logic\\_problems.pdf](https://www.ecs.soton.ac.uk/sites/www.ecs.soton.ac.uk/files/logic_problems.pdf). [Accessed 2020].
- [30] Android, “3.5 mm Headset Jack: Device Specification,” Android, [Online]. Available: <https://source.android.com/devices/accessories/headset/jack-headset-spec>. [Accessed 2020].
- [31] GDP-18, “Electronics Everywhere Application Walkthrough,” [Online]. Available: <https://www.youtube.com/watch?v=K6jnNEfbDwE&feature=youtu.be>.
- [32] Códice Software S.L., “gmaster - Git GUI,” Códice Software S.L., [Online]. Available: <https://gmaster.io/>. [Accessed 2020].
- [33] ar.to, “model\_viewer,” [Online]. Available: [https://pub.dev/packages/model\\_viewer](https://pub.dev/packages/model_viewer).
- [34] RS, “RS PRO 3.5 mm PCB Mount Composite Video Jack Socket, 4Pole,” RS Electronics, [Online]. Available: <https://uk.rs-online.com/web/p/jack-plugs-sockets/8051665/>. [Accessed 2020].

# 11 Appendix

<b>A ORIGINAL PROJECT BRIEF .....</b>	<b>100</b>
<b>B SPECIFICATION .....</b>	<b>101</b>
<b>C GITHUB BRANCH HISTORY .....</b>	<b>104</b>
<b>D RISK ASSESSMENT .....</b>	<b>106</b>
<b>E CLIENT FEEDBACK .....</b>	<b>108</b>
E.I.    CLIENT RESPONSE .....	108
E.II.   INITIAL E-MAIL .....	109
<b>F EDITED LABORATORY NOTES.....</b>	<b>110</b>
<b>G STOCK PHOTO HIGHLIGHTS .....</b>	<b>116</b>
<b>H APPLICATION INSTRUCTIONS .....</b>	<b>118</b>
<b>I HANDOVER DOCUMENT .....</b>	<b>127</b>
I.I.    INTRODUCTION AND PURPOSE OF DOCUMENT .....	127
I.II.   LIST OF FEATURES .....	128
I.III.  KNOWN BUGS .....	131
I.IV.   SOFTWARE INFORMATION AND SUGGESTED WORK .....	133
I.V.    HARDWARE INFORMATION AND SUGGESTED WORK .....	149
I.VI.   SUGGESTED CHANGES FOR SUPPORTING CONTENT.....	154
I.VII.  ADDITIONAL INFORMATION .....	155
I.VIII. PROJECT STRUCTURE .....	157

# A Original Project Brief

## School of Electronics and Computer Science Part IV Group Design Projects 2020/2021 Project Proposal

Note: The GDP is a substantial engineering design or feasibility study undertaken by a group of about four students. It carries 45 credits and, as a guideline, students should expect to spend about three quarters of their time on GDP related activities during the first semester in their fourth year. From 2020 the GDP is a 1 semester module.

<b>Project Title:</b>	A cross-platform signal generator/oscilloscope/logic analyser app
<b>Proposer(s):</b>	Geoff Merrett
<b>Research Group:</b>	Cyber Physical Systems
<b>Brief Summary of Project</b> (150-200 words):	
<p>The UK Electronics Skills Foundation (a UK charity) has been working with academics at the University of Southampton to develop educational teaching kits to help make electronics more visible to school/college students (<a href="http://www.ecs.soton.ac.uk/kits">www.ecs.soton.ac.uk/kits</a>). While these have been very well received, many schools (including those that the kits could have the most impact in) do not have access to electronic lab equipment, e.g. a signal generator, oscilloscope, logic analyser. This project will design, implement and test a cross-platform (at a minimum, Android and iOS) app to provide the functionality of such equipment on a mobile device, interfacing with the kits via the headphone/microphone socket. While apps with some functionality are already available on the market (e.g. <a href="https://play.google.com/store/apps/details?id=com.keuwl.functiongenerator">https://play.google.com/store/apps/details?id=com.keuwl.functiongenerator</a>), the customer is seeking a solution tailored specifically to the kits. If successful, there is the potential for the app to be distributed to hundreds of schools (and thousands of users) around the country.</p>	
<p>If the project has an <b>industrial customer</b> (recommended), please provide the customer's name, address,</p> <p>Stewart Edmondson (CEO) UK Electronics Skills Foundation North End House North End Ashton Keynes Wiltshire SN6 6QR</p>	
<p>Usually groups of 5 students will be allocated, however please identify the course(s) of study for students required for the project and if a project with mixed skills is required please identify how many students from each course?</p> <p>Majority Software Engineering or CS students. One or two Electronic Engineering students – provided they have ability/interests in software/app development would be useful however, seeing as the application of the software is in electronics (an app for virtual laboratory equipment).</p>	
Will this project require laboratory space? <b>Yes</b>	If yes for how many students (at one time)? 1
<p>What resources will be required:</p> <p><b>Supplies (please specify):</b> <b>Estimated Cost £</b></p> <ol style="list-style-type: none"> <li>May require access to a mobile device if don't already have access. These may already be loanable from ECS.</li> <li>May require license for Apple Developer Program (99USD)</li> </ol> <p>Is project viability dependent on additional funds or resources (e.g. provided by the customer?) No</p>	

Please return forms by email to [nrh@ecs.soton.ac.uk](mailto:nrh@ecs.soton.ac.uk) before Wednesday 24th June 2020. Include the text 'GDP Proposal 2020/21' in the subject line of your email.

## B Specification

### School of Electronics and Computer Science ELEC6200 MEng Group Design Project

#### Project Specification and Plan

GDP Number and Title: 18 - A cross-platform signal generator/oscilloscope/logic analyser app

Academic Supervisor(s): Geoff Merrett

Team Members: Eamonn Trim, Gavin Fish, Adel Hedayat, Leonardo Moreira, Fivos Gaitantzis

External Partner (including affiliation): Stewart Edmondson, UKESF

#### **Project Specification:**

**Problem:** Education of electronics at A-level or GCSE level is not mainstream, and sometimes all students receive are some experiments as part of the A-level Physics syllabus. An additional hurdle is that some Physics teachers do not feel comfortable enough with the field as to enthuse students to pursue electronics at further education. Even if teachers are enthused by the field, the lab equipment may either not be accessible, or be limited as to remove the possibility for students (kinaesthetic learners, in particular) to engage with the lesson.

**Solution:** We are aiming to produce a cross-platform application that can be used with the UKESF Music Mixer Kit (<https://ecs.soton.ac.uk/kits>) to provide A-level Physics teachers more tools enabling teaching electronics to students. This application will contain instructions relating to the electronics experiments found in A-level Physics syllabus to work with the Music Mixer Kit, whilst also providing companion tools for these experiments (that can also be used outside of these experiments). With most of the field being invisible to the naked eye, it is hard for young students to get a grasp on the interaction of electronic signals thus the application will visualise these to provide students with insight. The application will aid both teachers and students in explaining background theory of the experiments and the circuit boards included in the kits. More specifically with the Music Mixer Kit, users will be able to produce two individual mono-channel signals from the audio port of the device, whilst being able to use the microphone as an input (all to be displayed on the visualisation).

#### **Core Features:**

- Visualisation of an input signal from the microphone
  - 44.1kHz sample rate at 16-bit resolution
  - Ability to change the scale, relative phase to the generated signals and auto set
- Signal generator through the audio port
  - A maximum of 44.1kHz maximum voltages is 2V peak-to-peak at 16-bit resolution at loads of 100-600 Ohms impedance
  - Two individual mono-channel signals on the left and right audio channel
  - Sine, square, triangle waves, music samples as default options
  - Ability to change amplitude, relative phase, frequency
  - Can be individually turned off and on (if both are off, then music can be played)
  - To be directly visualisable alongside the input signal from the microphone
- Instruction sets for experiments
  - Targeted at students to follow (enabling simple lesson plans for teachers)
  - Covers all the material already available at (<https://ecs.soton.ac.uk/kits>)
  - Provides contextual background knowledge and theory about the experiment
- User interface
  - Targeted primarily at students to use, but also with teachers in mind
  - Links to UKESF & University of Southampton tools (e.g. [www.ecs.soton.ac.uk/kits](http://www.ecs.soton.ac.uk/kits))
  - Branding and colour scheme matching UKESF and Electronics Everywhere
  - Focusing on functional features to support experiments, without too many additional to avoid an overwhelming interface
  - Save/load slots to retain specific configurations

**Stretch Features:**

- User interface
    - Pinch zooms and edits for signals
    - Interactive pictures of boards to give information about operation
    - Addition instructional tasks to produce certain output signals (within tolerances) based on changing the circuit / signal configuration (providing the more adept students with a chance to explore more difficult application of physics, i.e. using these two output signals, and using the board, produce a signal that is a mix of the first two)
  - Logic analyser
    - Information on the board included within application
    - Instructions for the experiments on the logic analyser found on the website
    - Produce technical plan for implementing a logic analyser with the application (including hardware if required)
- 

**Project Plan:**

Using the specifications above, the tasks of the project were split up into the following stages:

- IOS and Android
  - Produce audio signals for left and right audio slots individually
  - Read in signal from microphone
  - Validate audio signals for left and right audio slots individually by using lab equipment
  - Validate reading in signals from microphone by using lab equipment
- Shared app development
  - Home screen with
    - Links to UKESF pages
    - Navigation layout for using visualiser / signal generator both separately and overlaid
    - Library of experiments instructions to launch
  - Design signal visualisation front end
  - Signal visualisation front end
  - Design signal generator front end
  - Signal generator front end
  - Implementation of core features
    - Save / load configurations
    - Auto set the visualisation
    - User interface branding
  - Instructions for all the experiments found on the website for the kits
- Testing
  - Testing the operation of the application will be handled internally throughout the project, with specific reference to the specification
  - Specific hardware testing will be carried out in a lab (subject to Covid-19 restrictions) to ensure performance matches the specification
  - At key points throughout the project, surveys will be sent (pending ethics approval) to gain insight into the requirements of end-users (teachers)
  - Testing in class is not possible within the time span of the project



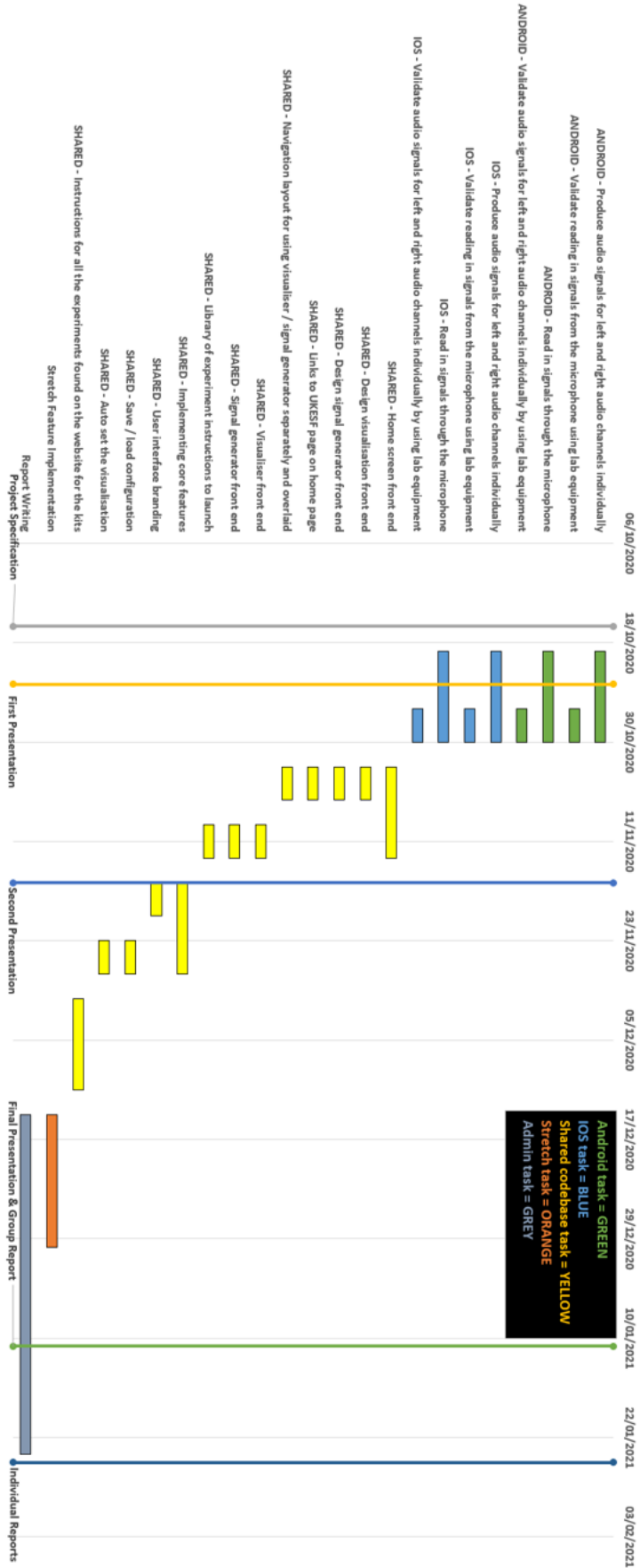


Figure 1: Gantt chart showing the project overview

# C GitHub Branch History

Graph	Description	Date	Author
	<b>Uncommitted changes</b>	2 Jan 2021 14:42	*
	<b>master Merge branch 'Eamonn-Integration'</b>	29 Dec 2020 16:31	MMRREE
	Eamonn-Integration updated tree and changed small wording bug	29 Dec 2020 16:30	MMRREE
	integrated adels updates	28 Dec 2020 12:21	MMRREE
	updating small bug fixes	20 Dec 2020 19:06	MMRREE
	Getting all the git history merged and aligned	16 Dec 2020 17:50	MMRREE
	gavins_changes Back behaviour for Logic and Arithmetic, and changed num of back presses before prompt	15 Dec 2020 18:14	gf4g17 <f
	Back button now returns you to home if you were just in kit and not selected anything	15 Dec 2020 17:54	gf4g17 <f
	Merge remote-tracking branch 'origin/Eamonn-Integration'	16 Dec 2020 17:41	MMRREE
	Merge remote-tracking branch 'origin/Leo-UI' into Eamonn-Integration	16 Dec 2020 17:40	MMRREE
	Leo-UI Update to lab pages	13 Nov 2020 17:10	Leo-More
	Merge remote-tracking branch 'origin/adding_settings_and_view_bar' into Eamonn-Integration	16 Dec 2020 17:36	MMRREE
	adding_settings_and_view_bar Implemented settings into bar for sig gen functionality	12 Nov 2020 19:30	gf4g17 <f
	A few more settings	12 Nov 2020 14:54	gf4g17 <f
	Added graph view functionality to both parts	12 Nov 2020 11:30	gf4g17 <f
	Started ne combined wave painter class, Added initial graph splitting functionality	12 Nov 2020 0:34	gf4g17 <f
	Functionality of view bar added	9 Nov 2020 19:55	gf4g17 <f
	updated to be immutable as per flutter problem	16 Dec 2020 17:27	MMRREE
	Merge remote-tracking branch 'origin/Leo-Tweaks' into Eamonn-Integration	16 Dec 2020 17:25	MMRREE
	Leo-Tweaks Update for tablet UI scaling	15 Dec 2020 20:44	Leo-More
	All 10 labs for logic	15 Dec 2020 18:30	Leo-More
	updated buffers to work of buffer size	16 Dec 2020 17:22	MMRREE
	Merge branch 'Eamonn-Integration' of git.soton.ac.uk:ect1u17/gdp-18 into Eamonn-Integration	15 Dec 2020 17:25	MMRREE
	Changed behaviour of back arrow in top bar to include optional current index variable	15 Dec 2020 17:06	gf4g17 <f
	updating wave painter to do calculation for number of buffers to keep based on buffer size	15 Dec 2020 17:24	MMRREE
	Aligned Logic and Arithmetic name to the rest	15 Dec 2020 11:52	gf4g17 <f
	renamed file properly and added vertical requirement for whole application on launch	14 Dec 2020 18:59	MMRREE
	removed shadows from practical page	14 Dec 2020 17:45	MMRREE
	changing app to use lower case .png	14 Dec 2020 17:27	MMRREE
	Merge branch 'Leo-Tweaks' into Eamonn-Integration	14 Dec 2020 17:20	MMRREE
	import update	13 Dec 2020 16:22	Leo-More
	Expansiontile for solutions in lab	13 Dec 2020 16:20	Leo-More
	Fixed lab theme issue	13 Dec 2020 14:53	Leo-More
	Logic labs, fixed lab container issue, added labs to practical button not yet implemented	13 Dec 2020 14:50	Leo-More
	changed file formats to be lower case	14 Dec 2020 17:08	MMRREE
	fully removing the shadows from the boxes	14 Dec 2020 17:02	MMRREE
	Merge branch 'Fivos-Final-Fixes' into Eamonn-Integration + removing the shadow around boxes that are undeded	14 Dec 2020 17:02	MMRREE
	Fivos-Final-Fixes Replace edit_channel.dart	14 Dec 2020 15:33	fg1g17 <f
	Replace mixer_practical.dart	14 Dec 2020 15:33	fg1g17 <f
	Replace model.dart	14 Dec 2020 15:32	fg1g17 <f
	Replace main.dart	14 Dec 2020 15:31	fg1g17 <f
	Replace controller.dart	14 Dec 2020 15:31	fg1g17 <f
	Replace AppDelegate.m	14 Dec 2020 15:30	fg1g17 <f
	Merge branch 'Fivos-Bug-Fixes' into Eamonn-Integration (4 days old)	14 Dec 2020 16:57	MMRREE
	Fivos-Bug-Fixes Replace AppDelegate.m	10 Dec 2020 14:38	fg1g17 <f
	Replace mixer_practical.dart	10 Dec 2020 14:37	fg1g17 <f
	Replace layout_selector.dart	10 Dec 2020 14:37	fg1g17 <f
	Replace settings_bar_scope.dart	10 Dec 2020 14:36	fg1g17 <f
	Replace layout_selector.dart	10 Dec 2020 14:36	fg1g17 <f
	Replace parameter_tabs.dart	10 Dec 2020 14:35	fg1g17 <f
	Replace edit_channel.dart	10 Dec 2020 14:34	fg1g17 <f
	Replace controller.dart	10 Dec 2020 14:27	fg1g17 <f
	Merge branch 'Eamonn-Integration' into 'master'	7 Dec 2020 18:39	ect1u17 <
	Merge branch 'Eamonn-Integration' of git.soton.ac.uk:ect1u17/gdp-18 into Eamonn-Integration	14 Dec 2020 14:55	MMRREE
	Addition to music mixer settings	11 Dec 2020 23:57	gf4g17 <f
	updated dependencies and package name to match flutter requirements	14 Dec 2020 14:55	MMRREE
	added custom icons for layout selected, removed deprecated updateAttributes and cleared up unprocessed mic	7 Dec 2020 18:38	ect1u17 <
	Bug fix for null dark mode and oscilloscope setting	7 Dec 2020 13:45	gf4g17 <f
	changes made based on suggestions from Stew and Geoff on Friday	6 Dec 2020 19:40	ect1u17 <
	integrated adels updated files (logic and music mixer	3 Dec 2020 19:39	ect1u17 <
	updated close icon	3 Dec 2020 18:41	ect1u17 <
	added adels integration, theming and back buttong on graph page	3 Dec 2020 18:35	ect1u17 <
	integrated adels interactive image for music mixer	3 Dec 2020 13:24	ect1u17 <
	restructure to include renaming files and to make it more clear about the segration of tasks for each view	2 Dec 2020 23:21	ect1u17 <
	updated settings to be included and linked up	2 Dec 2020 19:59	ect1u17 <
	Merge remote-tracking branch 'origin/Leo-From-Master' into Eamonn-Integration	2 Dec 2020 19:28	ect1u17 <
	Leo-From-Master New images	2 Dec 2020 18:48	Leo-More
	Widget tweaks	2 Dec 2020 17:32	Leo-More
	updated some theming on practical page, and edited zoom to work better	2 Dec 2020 19:23	ect1u17 <
	updated to include foivos' code and to use the app theming in the practical page (still WIP)	2 Dec 2020 18:10	ect1u17 <
	Merge remote-tracking branch 'origin/Fivos-Save-Load-Slots' into Eamonn-Integration	2 Dec 2020 16:51	ect1u17 <
	Fivos-Save-Load-Slots Fixed a bug.	2 Dec 2020 16:28	fg1g17 <f
	Updated pubspec.yaml to include SharedPreferences flutter dependency.	2 Dec 2020 16:21	fg1g17 <f
	Updated Side Bar Selection to include the Save/Load slots button.	2 Dec 2020 16:20	fg1g17 <f
	Updated controller to include all back end functions for the Save and Load (in the bottom of the file)	2 Dec 2020 16:19	fg1g17 <f
	Added Context to the side bar selection.	2 Dec 2020 16:18	fg1g17 <f
	Replace AppDelegate.m so that it compiles. Not yet finished to support the new updateValue functionality.	2 Dec 2020 16:17	fg1g17 <f
	A file containing all the popups used by the Save/Load functionality.	2 Dec 2020 16:16	fg1g17 <f
	Added Custom Dialog Boxes for Manually Settings Frequency, Amplitude & Offset matching the UKESF theme & branding	2 Dec 2020 16:15	fg1g17 <f
	Added important model variables for Save/Load Slots.	2 Dec 2020 16:14	fg1g17 <f
	updated to use flex boxes on tab sig gen settings	2 Dec 2020 16:03	ect1u17 <
	light and dark mode addition (unconnected so far)	2 Dec 2020 13:00	ect1u17 <
	Merge remote-tracking branch 'origin/Leo-From-Master' into Eamonn-Integration	2 Dec 2020 13:00	ect1u17 <
	UI update for branding	2 Dec 2020 0:04	Leo-More
	updated graphing page to include trigger option, to be more readable, and to include theming for colour and origin setti	1 Dec 2020 17:16	ect1u17 <
	updated controller to update attributes on changing tab and tried to update native sides to use the functions	1 Dec 2020 1:09	ect1u17 <
	refactoring code, updating some bugs, and aligning some UI	30 Nov 2020 23:38	ect1u17 <
	updated integration with Leos	30 Nov 2020 16:15	ect1u17 <
	Merge remote-tracking branch 'origin/Leo-From-Master' into Eamonn-Integration	30 Nov 2020 16:15	ect1u17 <
	Update to lab pages and UI design	28 Nov 2020 16:22	Leo-More

Graph	Description	Date	Author
	triggering experimental build, not yet finished	28 Nov 2020 18:04	ect1u17 <
	removed the "warnings" dart gives	28 Nov 2020 15:29	ect1u17 <
	updated some GUI in combined graphs	28 Nov 2020 15:25	ect1u17 <
	connecting oscilloscope scale to other graphs on the same pages, bug on 1x zoom at the moment	27 Nov 2020 20:27	ect1u17 <
	Merge remote-tracking branch 'origin/gavins_changes' into Eamonn-Integration	27 Nov 2020 19:42	ect1u17 <
	Made some aesthetic tweaks to signal generator settings	27 Nov 2020 17:30	gf4g17 <g
	Added output buttons for left/right, removed master toggle, added back end functions	27 Nov 2020 14:08	gf4g17 <g
	edited tabbed style for selecting channel to edit	26 Nov 2020 17:41	gf4g17 <g
	Added tabbed settings at the bottom	26 Nov 2020 16:14	gf4g17 <g
	Fixed merge issue where text couldn't be submitted	26 Nov 2020 12:04	gf4g17 <g
	Merge branch 'Eamonn-Integration' into gavins_changes	25 Nov 2020 22:39	gf4g17 <g
	committing changes to wave painter, triggering in progress	27 Nov 2020 19:40	ect1u17 <
	Merge branch 'Leo-From-Master' into Eamonn-Integration	27 Nov 2020 11:28	ect1u17 <
	UKESF branding for home page	26 Nov 2020 22:47	Leo-More
	small updates for performance	27 Nov 2020 11:19	ect1u17 <
	small updates to fix scaling	25 Nov 2020 22:30	ect1u17 <
	Merge remote-tracking branch 'origin/gavins_changes' into Eamonn-Integration	25 Nov 2020 21:33	ect1u17 <
	Finished functional input parameter pop up	25 Nov 2020 17:53	gf4g17 <g
	merging gavins newest changes	25 Nov 2020 21:32	ect1u17 <
	Merge remote-tracking branch 'origin/gavins_changes' into Eamonn-Integration	25 Nov 2020 17:58	ect1u17 <
	Working text field for frequency, migrated some functions to control	25 Nov 2020 14:24	gf4g17 <g
	started adding enter your own frequency feature	25 Nov 2020 11:14	gf4g17 <g
	graphing uses model colors	21 Nov 2020 17:27	gf4g17 <g
	Slider colors based on which you are editing	21 Nov 2020 15:09	gf4g17 <g
	Code cleanup and some gui changes	21 Nov 2020 14:32	gf4g17 <g
	Slight code cleanup	21 Nov 2020 13:52	gf4g17 <g
	Opens settings when view changed	21 Nov 2020 12:33	gf4g17 <g
	Graph template UI change	20 Nov 2020 16:23	gf4g17 <g
	updated to have a set resolution for the graph and trying to fix play and listen automatically on app reload (buggy)	25 Nov 2020 17:49	ect1u17 <
	merges and bug fixing to get rid of warning and errors in the code and in the console	25 Nov 2020 14:50	ect1u17 <
	Merge branch 'Leo-From-Master' into Eamonn-Integration	25 Nov 2020 14:01	ect1u17 <
	Background lab Image	24 Nov 2020 16:41	Leo-More
	Update to GUI to look nicer as discussed in meeting	24 Nov 2020 15:48	Leo-More
	Merge branch 'master' into Eamonn-Integration	25 Nov 2020 13:59	ect1u17 <
	SaveLoad-Branch-Fivos Update microphone_input_stream.dart to not await for permission status when its running o	24 Nov 2020 17:59	fg1g17 <f
	Added a functionality for iOS to turn on the recording unit when startListening() is called and stop the recording unit wher	24 Nov 2020 17:56	fg1g17 <f
	- Optimized AppDelegate header for full functionality (oscilloscope & signal generator) - Added Missing Wave-Types (Tri	24 Nov 2020 17:53	fg1g17 <f
	- Optimized AppDelegate header for full functionality (oscilloscope & signal generator)	24 Nov 2020 17:52	fg1g17 <f
	Merge branch 'Eamonn-Integration' into 'master'	15 Nov 2020 14:11	ect1u17 <
	Merge branch 'Eamonn-Integration' into 'master'	15 Nov 2020 13:55	ect1u17 <
	WIP improvements to the GUI and graphing	25 Nov 2020 12:53	ect1u17 <
	Updated some other bugs and features for GUI	20 Nov 2020 18:54	ect1u17 <
	Bug fixes for graphing display	20 Nov 2020 16:48	ect1u17 <
	merged LeoUI to integration	15 Nov 2020 14:10	ect1u17 <
	merge Leo's branch to integration	15 Nov 2020 14:06	ect1u17 <
	Updated GUI elements and removed some deprecated code	15 Nov 2020 13:55	ect1u17 <
	Updated GUI elements and removed some deprecated code	13 Nov 2020 17:47	ect1u17 <
	Integrated with Gavins updated code	12 Nov 2020 21:53	ect1u17 <
	Integrated the first version of Gavin's code with the MVC	12 Nov 2020 18:13	ect1u17 <
	Mostly integrated gavins WIP stuff onto the integration page	12 Nov 2020 15:55	ect1u17 <
	Added some intergration from Gavins part, will be adding more but checking out the branch seems to get rid of some of r	12 Nov 2020 15:13	ect1u17 <
	Updated to fix some bugs	12 Nov 2020 14:52	ect1u17 <
	Updated the current integration to MVC	11 Nov 2020 13:13	ect1u17 <
	Updated mic input to work with android, needs to be tested on ios	9 Nov 2020 19:52	ect1u17 <
	Updated some integration	9 Nov 2020 15:42	ect1u17 <
	Updated ios integration to work with the functions (hopefully). Also worked on splitting the WIP gui into a MVC methodol	8 Nov 2020 15:47	ect1u17 <
	Updated ios integration to work with the functions (hopefully). Also worked on splitting the WIP gui into a MVC methodol	8 Nov 2020 15:33	ect1u17 <
	Went back to older version of AppDelegate supporting updateAttribute as opposed to multiple functions in order to change	8 Nov 2020 0:58	fg1g17 <f
	- More compact code - Added Output Bar & Edit Bar - Fixed some issue with async functions and iOS.	8 Nov 2020 0:55	fg1g17 <f
	Replaced case with if statement (Strings not supported for case in objective C)	7 Nov 2020 18:27	fg1g17 <f
	Replace AppDelegate.h with updated one.	7 Nov 2020 18:26	fg1g17 <f
	Added permissions for Microphone on iOS.	7 Nov 2020 18:25	fg1g17 <f
	Updated ios integration to work with the functions (hopefully). Also worked on splitting the WIP gui into a MVC methodol	7 Nov 2020 16:38	ect1u17 <
	Updated to have UKESF logo and be called EToolkit.	6 Nov 2020 21:07	ect1u17 <
	Updated back button operation on music mixer page	6 Nov 2020 17:21	ect1u17 <
	updated lab page	6 Nov 2020 13:44	ect1u17 <
	Readme Change	6 Nov 2020 12:28	ect1u17 <
	Merge branch 'master' of git.soton.ac.uk:ect1u17/gdp-18	6 Nov 2020 12:22	ect1u17 <
	Update README.md	6 Nov 2020 11:26	ect1u17 <
	Update README.md	6 Nov 2020 11:24	ect1u17 <
	Update README.md	6 Nov 2020 11:23	ect1u17 <
	Update README.md	6 Nov 2020 11:23	ect1u17 <
	Update README.md	6 Nov 2020 11:20	ect1u17 <
	Merge branch 'Leo-UI' into 'master'	6 Nov 2020 11:18	ect1u17 <
	Lab Slider and lab pages update	5 Nov 2020 12:08	Leo-More
	Merge branch 'Leo-UI' into 'master'	4 Nov 2020 14:58	ect1u17 <
	Simplified MVC structure	2 Nov 2020 12:04	Leo-More
	Implemented brower link	30 Oct 2020 15:49	Leo-More
	UI Update with MVC initial architecture	29 Oct 2020 19:33	Leo-More
	new file: _supporting_documents/GanntChart.png	6 Nov 2020 12:22	ect1u17 <
	modified: README.md	6 Nov 2020 12:22	ect1u17 <
	Merge branch 'Leo-UI' into 'master'	28 Oct 2020 15:22	ect1u17 <
	UI Initial Update	26 Oct 2020 16:50	Leo-More
	Default flutter project commit	21 Oct 2020 0:12	Leo-More
	Initial commit	17 Oct 2020 18:28	ect1u17 <

# D Risk Assessment

## Pre-Project Risk Assessment

**Imp.:** A score out of ten for how detrimental the risk would impact the project

**Prob.:** The estimated probability of the risk occurring

**Risk:** The calculated risk that this item would have on the project

ID	Risk	Hazard	Imp.	Prob.	Risk	Mitigate By
<b>Technical</b>						
T1	Project work gets corrupted or lost	Work done is lost and may have to start over	9	0.1	0.9	Save work frequently. Use of Git and Microsoft Teams to save code and reports
T2	Not sure how to implement a certain aspect of the project plan	Delay on the project implementation and therefore the entire project	7	0.5	3.5	Tell project manager and ask supervisors for help. Continue working on other parts of the project. Do not set unreachable goals in the project specification.
T3	A part of the sprint is not finished	Delay on the project	5	0.5	2.5	Move that part to the next sprint and complete it by the end of the next sprint.
T4	No one in the group has either an IOS or no one in the group has an android device	Harder to test on other system reliably	7	0.1	0.7	Ask supervisor or university to borrow a physical device or use an emulator.
<b>Personal</b>						
P1	One of the group members travels to a different country	Impossible for all member to meet face-to-face making it harder to discuss topics	3	0.3	0.9	Contact project manager and supervisor, and use Microsoft Teams, Trello, Facebook Messenger and any other forms of online contact necessary.
P2	One of the group members goes out of Southampton	Unlikely for all members to meet up face-to-face making it harder to discuss topics	3	0.5	1.5	Use Microsoft Teams, Trello, Facebook Messenger and any other forms of online contact necessary
P3	Work from other modules is hard to manage	Harder to complete tasks assigned with a lot of work from other modules	5	0.3	1.5	Good time management with other modules as well as project. Project manager assign tasks that are achievable by members.

External						
E1	Ethics application review for questionnaire gets denied.	Cannot get feedback from teachers who have already used the kits	6	0.1	0.6	Creating the application form early on and sending a draft to supervisor
E2	Ethics application takes longer than expected to be accepted.	Feedback from teachers is received late resulting in possible project delays	5	0.3	1.5	Applying for the ethics application early
E3	Cannot make a booking to meet up in person	Cannot discuss topics in person may make it harder to understand information	2	0.9	1.8	Use Microsoft Teams, Trello, Facebook Messenger and any other forms of online contact necessary
E4	There is a second UK lockdown due to Covid-19 during the project	Cannot meet in person to discuss topics. Increased stress staying in lockdown	5	0.7	3.5	Use Microsoft Teams, Trello, Facebook Messenger and any other forms of online contact necessary
E5	Cannot meet up face to face with client	Harder to understand what the client wants	3	0.8	2.4	Use Microsoft Teams, Trello, Facebook Messenger and any other forms of online contact necessary
E6	Client or supervisor does not respond to emails or other forms of contact	Harder to understand what the client wants and progress with the project	8	0.1	0.8	Try other forms of contact such as by phone or contact the university
E7	Cannot make a booking in labs to test hardware due to restrictions	Harder to test hardware for the project, including microphone and kit interaction	3	0.5	1.5	Use hardware accessible at home which should be enough for tests. Contact project supervisor if necessary.
E8	University closes due to a UK lockdown.	Work on project may be suspended for the foreseeable future or may have to continue studies online.	7	0.1	0.7	Follow project supervisor's advice and use Microsoft Teams and email to stay in contact with each other.

# E Client Feedback

## E.i. Client Response

Hi Eamonn,

Many thanks for forwarding the video. It is very good; nice and clear and Gavin should be commended for his measured delivery.

As requested, here is my feedback on the project.

### Technical work:

- Developing an app was not especially innovative, per se. However, the approach showed good innovative, the execution was done really well and the design solution was cleverly thought through.
- I didn't see the test plan or test spec, so it is difficult for me to comment about the thoroughness of the testing.
- I was impressed by the professional approach by the group. Their work was well planned and organised. I was very pleased with the outcome, which showed that the group understood both the context and the design brief.

### Communication:

- The Group leader (Eamonn) provided excellent weekly updates. Overall, communications was a strength. The group presentations were effective; well put together and professionally delivered.
- My impression is that a lot has been achieved in a short space of time. I was particularly pleased with the Group's (positive) response to the comments and feedback on their initial design brief. It was really good to see our comments being acknowledged and acted upon.
- My sense is that the group were really well-motivated and very organised; they seemed to work well together. My main engagement was with Eamonn and Gavin, so it is difficult for me to comment too much on the contribution of the other group members. Nevertheless, the amount of work undertaken to successfully complete a challenging project shows that everyone must have made a worthwhile contribution.

### Achievement of the project:

- In its current state, do you think that the application is:
  - o *Ready to be used*
  - o *Worth of being used*
  - o *Probably worth using*
  - o *Possibly worth using*
  - o *Further work needed*
  - o *Major work needed*
- With knowledge of the specification, and the outcome of the project, in your view, did the project:
  - o *Meet all goals*
  - o *Meet all major goals*
  - o *Meet most of the major goals*
  - o *Meet some of the goals*
  - o *Meet a few goals*

Best regards

Stew

## E.ii. Initial E-mail

Dear Stew,

First of all, thank you for your continued support throughout the project.

Hopefully, this feedback will come after you have seen the project handover documents and the [application walkthrough video](#).

I just wanted to ask some questions to understand any areas we handled well during this project and any areas that we might be able to improve on for our future experience.

To be clear, these questions intend to help you give constructive feedback and help support us in providing good evidence for the group report. We will use your comments (positive or negative) in the report to qualify our achievements.

Some questions relate to the criteria to gauge if you are satisfied with the project's outcome; thus, there is some specific wording to choose from (these are highlighted in *italics*).

### Technical work:

- Do you think that the work we have done is *innovative in its approach*?
- Do you think that the project was *thoroughly tested*?
- Are you *satisfied or impressed* with the outcome of the project?

### Communication:

- Did you feel as though you had a good overview of what was happening throughout the project?
- Did you think the team managed the time well and considered your feedback when working on the project?
- Did the team conduct themselves professionally, clearly and on time?

### Achievement of the project:

- In its current state, do you think that the application is:
  - o *Ready to be used*
  - o *Worth of being used*
  - o *Probably worth using*
  - o *Possibly worth using*
  - o *Further work needed*
  - o *Major work needed*
- With knowledge of the specification, and the outcome of the project, in your view, did the project:
  - o *Meet all goals*
  - o *Meet all major goals*
  - o *Meet most of the major goals*
  - o *Meet some of the goals*
  - o *Meet a few goals*
- Did you feel that the project was either a *very challenging project* or a *challenging project*?

Again, these questions help guide towards providing feedback that can help support arguing for the criteria. However, if you have any other feedback you would like to give us about the project that doesn't fit within these questions (such as if we were to repeat the project anything you would like changed), please feel free to do so.

Kind regards,  
Eamonn

## F Edited Laboratory Notes

ECS Electronic Engineering Kits

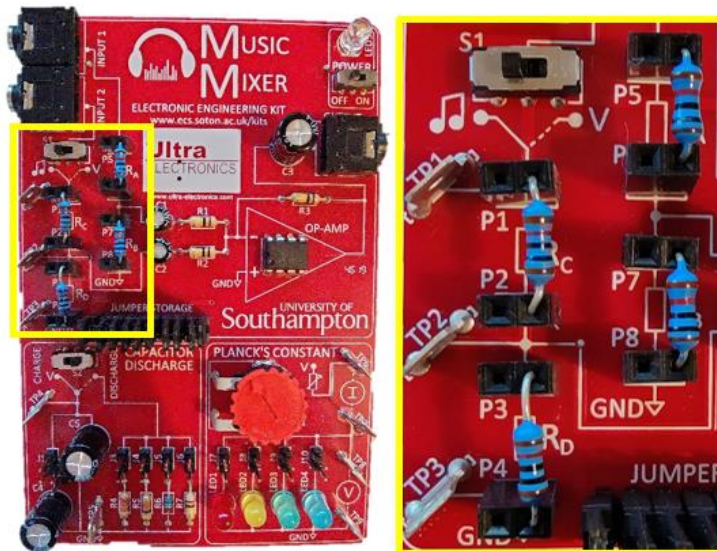
A-Level Physics: Music Mixer Kit

MM - 5

### 2.2 Potential dividers (AC)

Potential dividers can also be used to divide an AC signal. We are now going to connect the Music Mixer board to a phone running the Electronics Everywhere App.

Plug one  $10\text{k}\Omega$  resistor between P1 and P2 (i.e.  $R_C = 10\text{k}\Omega$ ), one between P3 and P4, one between P5 and P6, and one between P7 and P8, so that  $R_A = R_B = R_C = R_D = 10\text{k}\Omega$ . Make sure switch S1 is in the left “music” position.

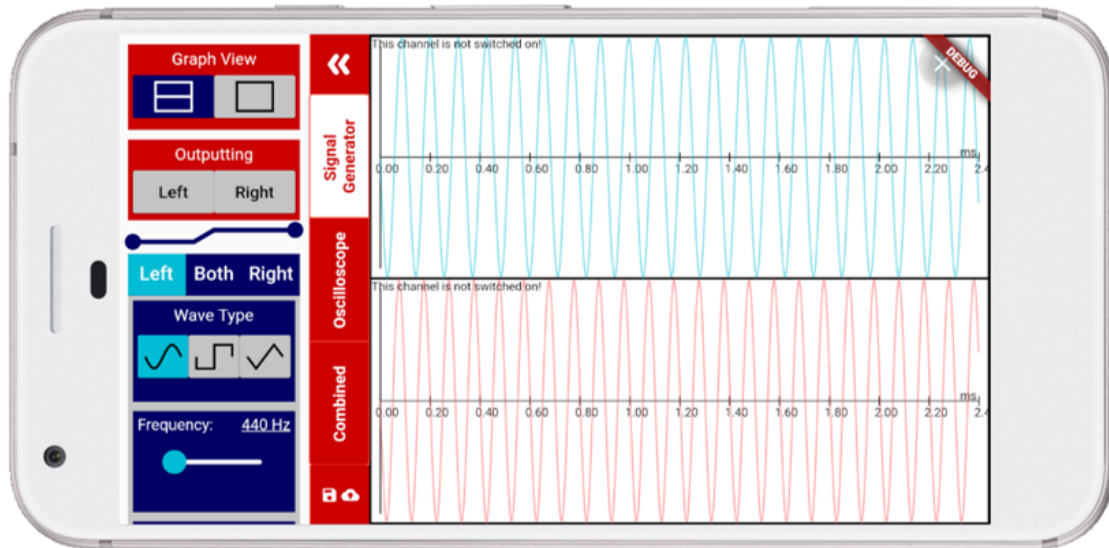


Turn the music mixer board off. Plug the “airplane” splitter into the inputs of the music mixer board. Connect the “airplane” splitter to the headphones part of the splitter (usually green). Connect the output of the music mixer board to the microphone part of the aux splitter (usually shown in pink). Connect the male end of the splitter into a phone’s auxiliary port. The setup should look something like this:





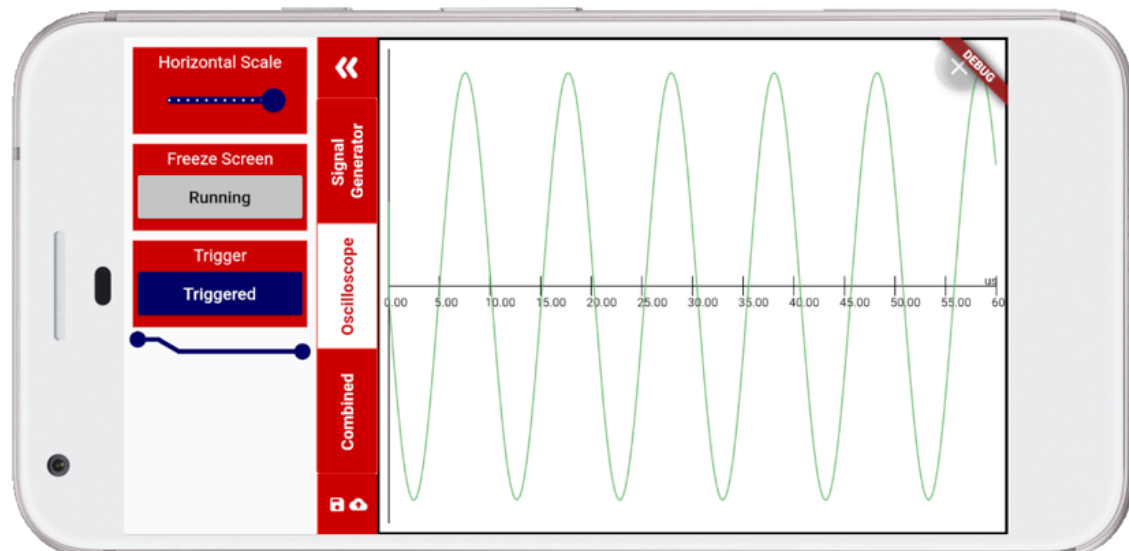
Open the Electronics Everywhere application and navigate 'home screen' -> 'music mixer' -> 'practical' and set the volume on your phone to about 50%. Turn the music mixer board on.



In the "signal generator" view, go to the left and right tabs and set the parameters as shown in the table below. Turn the left and right signals on by pressing on the "outputting" buttons.

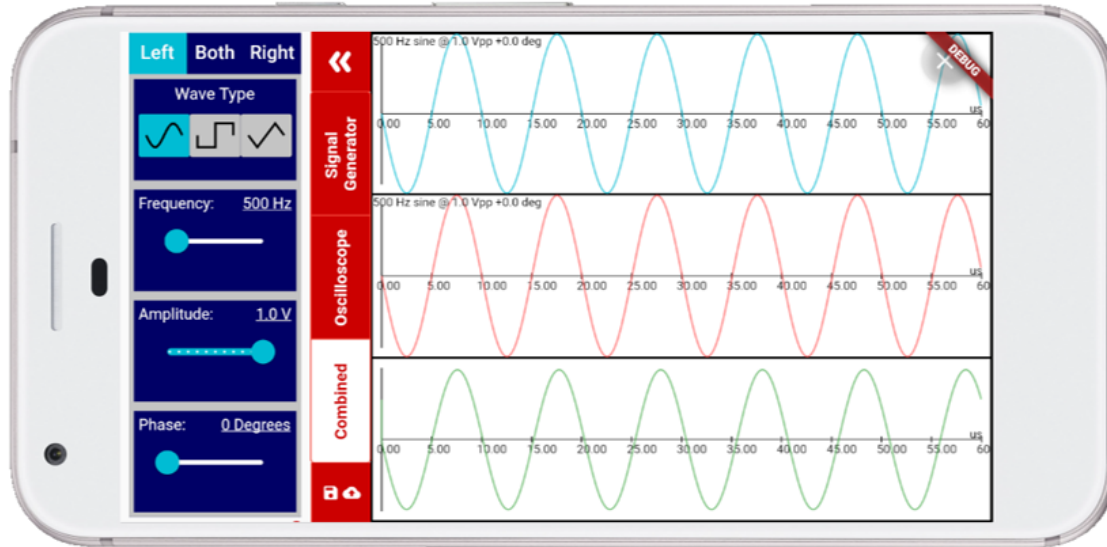
	Left	Right
<b>Frequency / Hz</b>	500	500
<b>Amplitude</b>	1.0	1.0
<b>Phase / Degrees</b>	0	0

Select the "oscilloscope" view and observe the output waveform. You can trigger the wave by pressing the button in the trigger section. You will see something like this:



Note: if the phone is not picking up any signal from the microphone, turn the music mixer board off for 10 seconds and then turn it back on again.

Now select the “combined” view to see both input signals to the music mixer board and the output signal all on one screen. You will see something like this:



Note: you can change the graph layout at the top of the parameter menu to see the signals in different graph configurations.

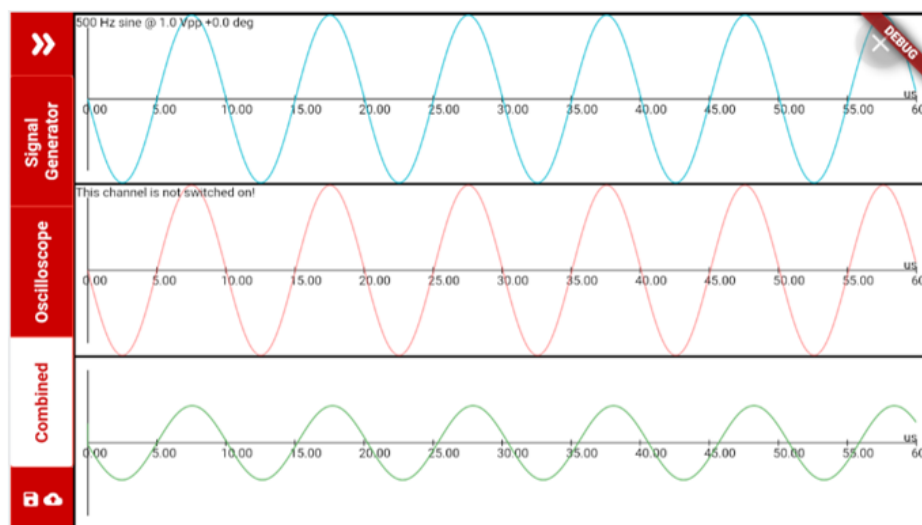
Select the tab to edit the right signal while still in the “combined” view. Change the phase of the signal by moving the phase slider. You can also manually set this value by pressing on the underlined value and inputting a desired value. Change the phase of this signal for the following values and note the behaviour: 0, 90, 180, 270, 360. Is this what you expect to happen?

Shown below are some images that represent different configurations of the signals and their phases. Try and re-create these using the app.

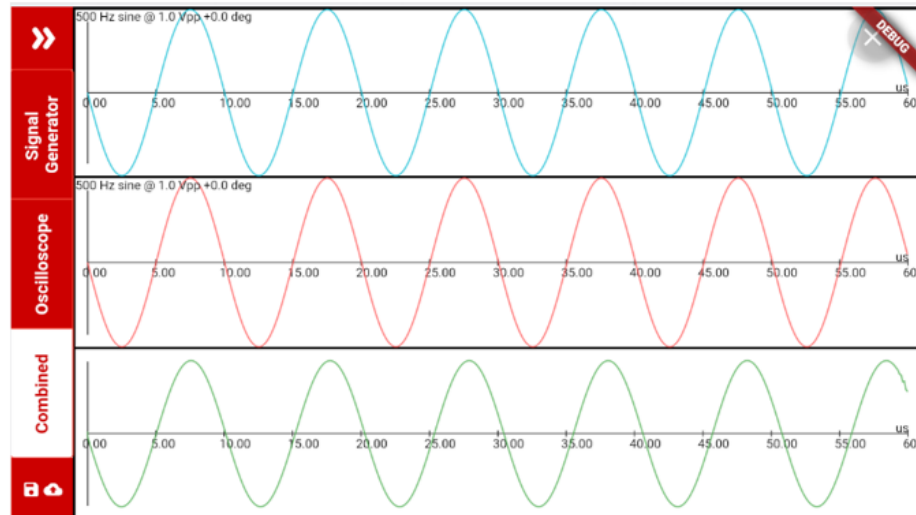
Note: you can change the scale of the “combined” or “oscilloscope” view using the “horizontal scale” slider.

Which of these illustrate constructive or destructive interference?

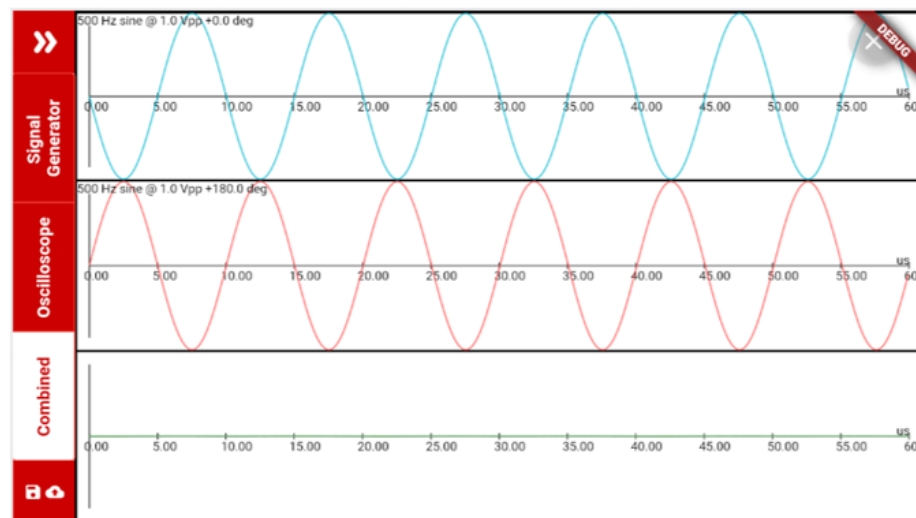
- (A)
- Left 0°
- Right OFF



(B)  
Left  $0^\circ$   
Right  $0^\circ$

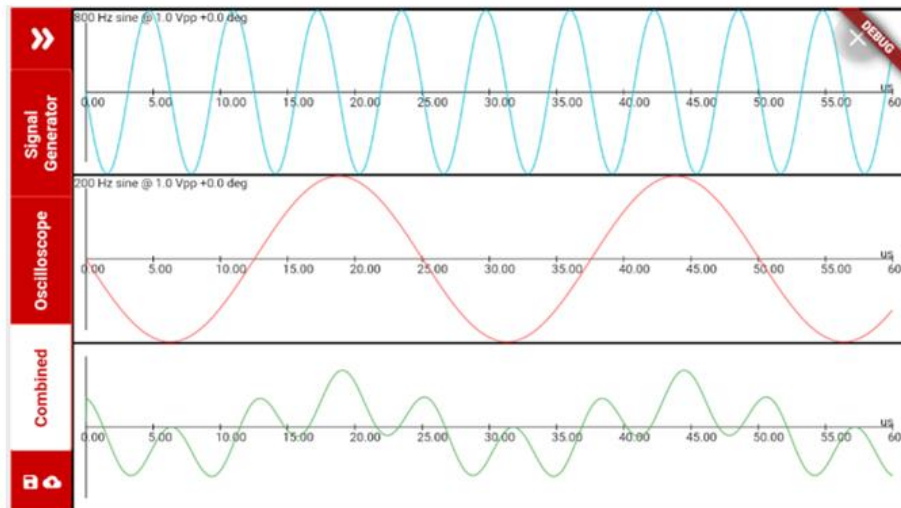


(C)  
Left  $0^\circ$   
Right  $180^\circ$

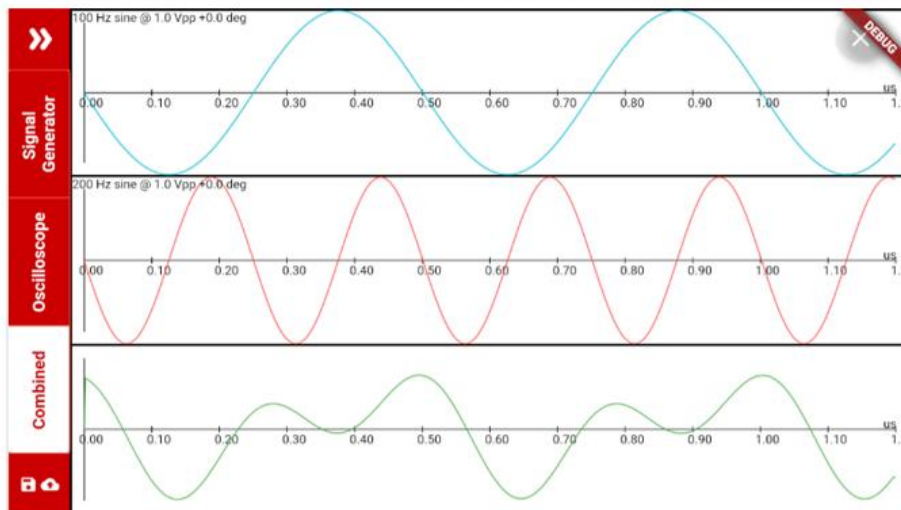


The op-amp circuit can also be used to mix different frequency signals. Adjust the phase of both signals back to zero, and then adjust the frequency of the left and right signals to the values stated below.

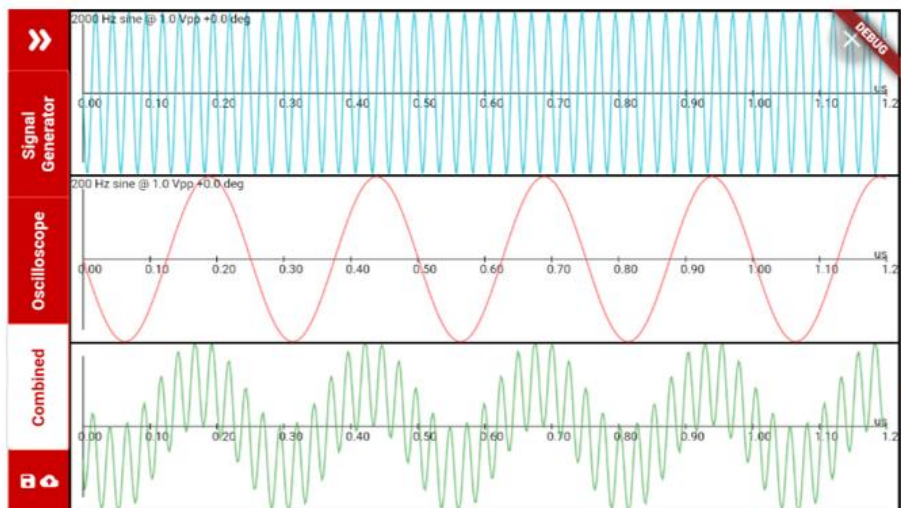
(D)  
Left 800Hz  
Right 200Hz



(E)  
Left 100Hz  
Right 200Hz

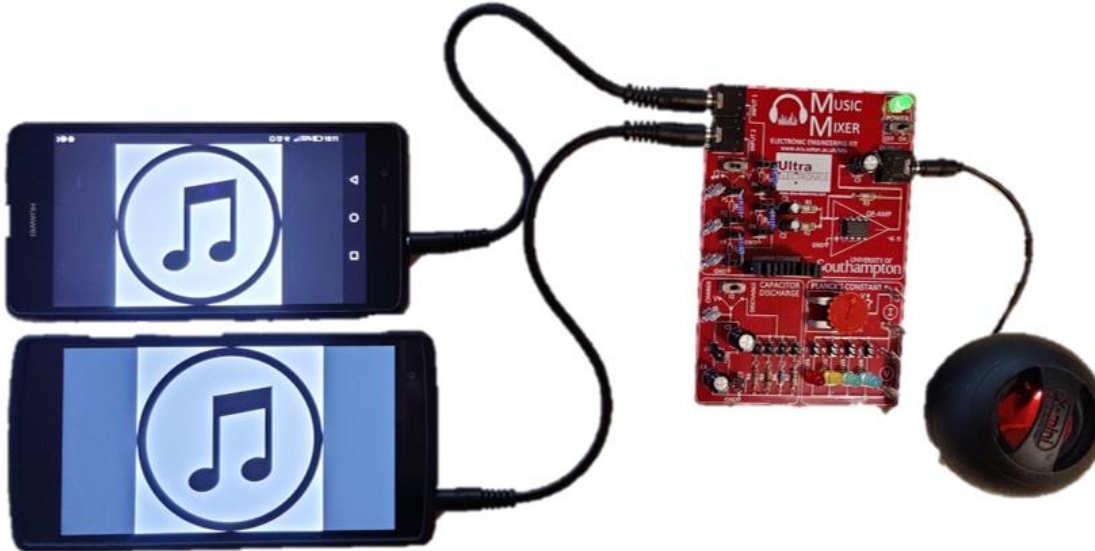


(F)  
Left 2000Hz  
Right 200Hz



### 2.3 Music Mixing

The music mixer board can also be used to mix 2 individual music sources together. Keep the resistors in the music mixer as they were setup in Section 2.2. Unplug the splitters and connect the board in the following way:



The speaker can be substituted with earbuds or headphones. Play some music from both phones and listen to the output. Is this what you expected it to sound like?

Unplug resistor  $R_D$  and replace it with an LDR. Cover the LDR and listen to the change in the output. How did this affect the music that is going through Input 2?

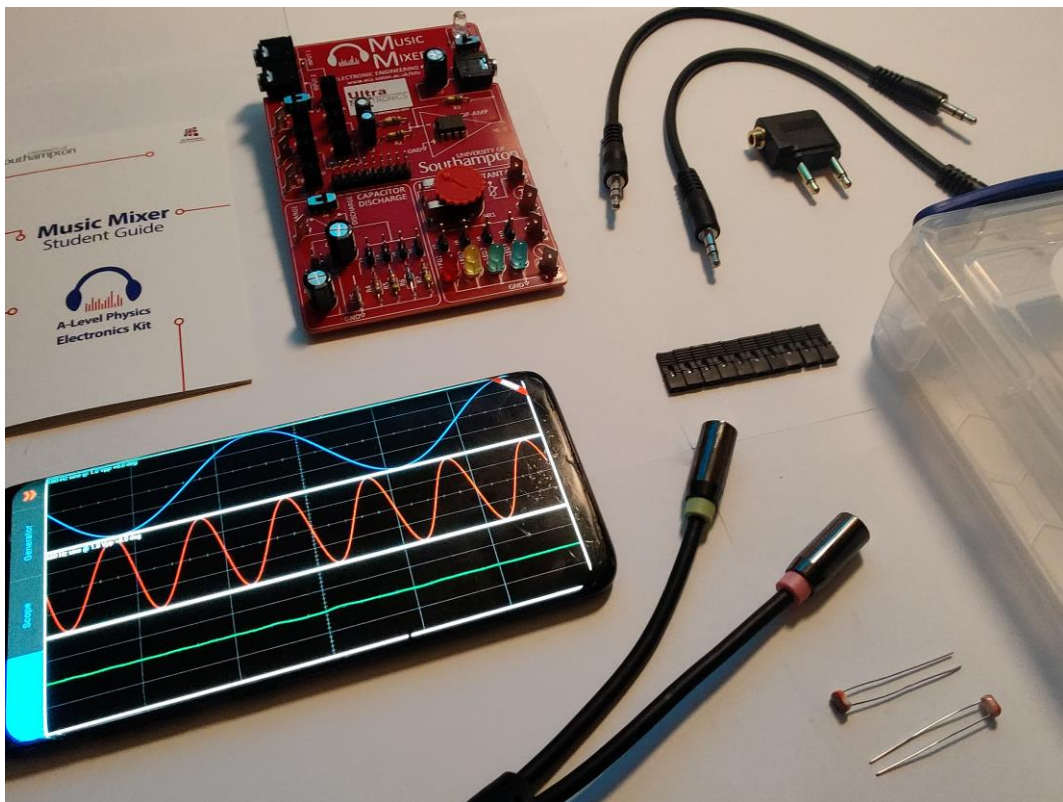
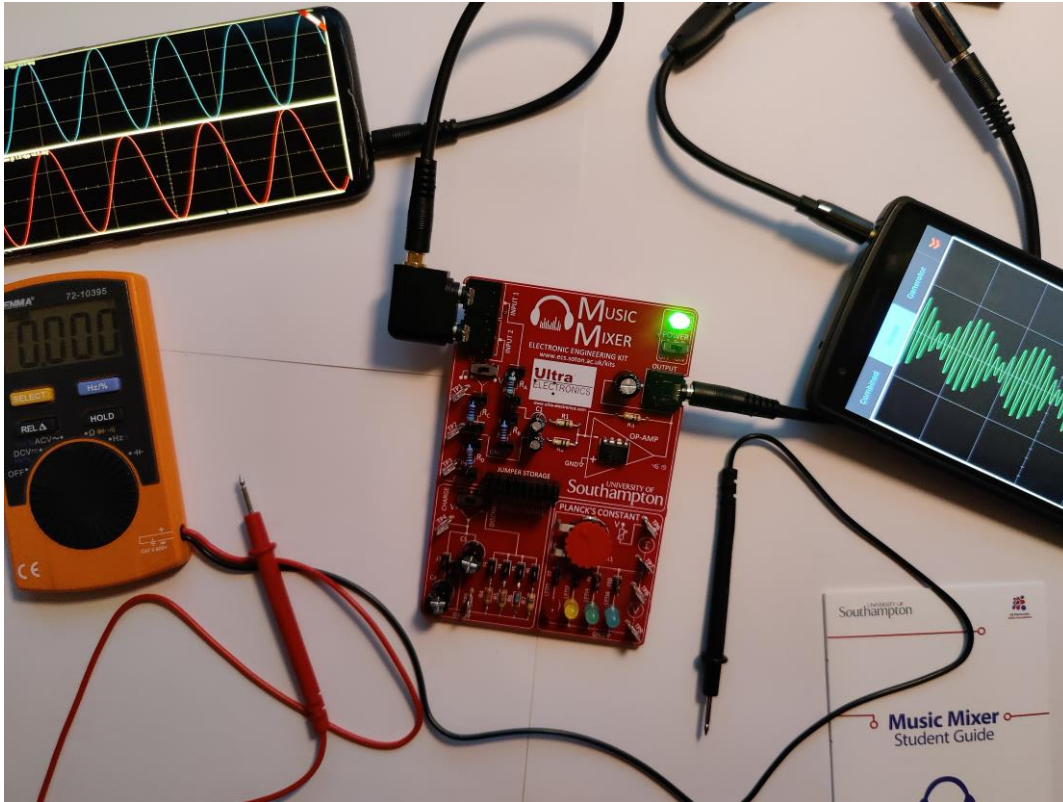
Shine a light using a phone's flashlight and listen to the change in the output. How did this affect the music that is going through Input 2?

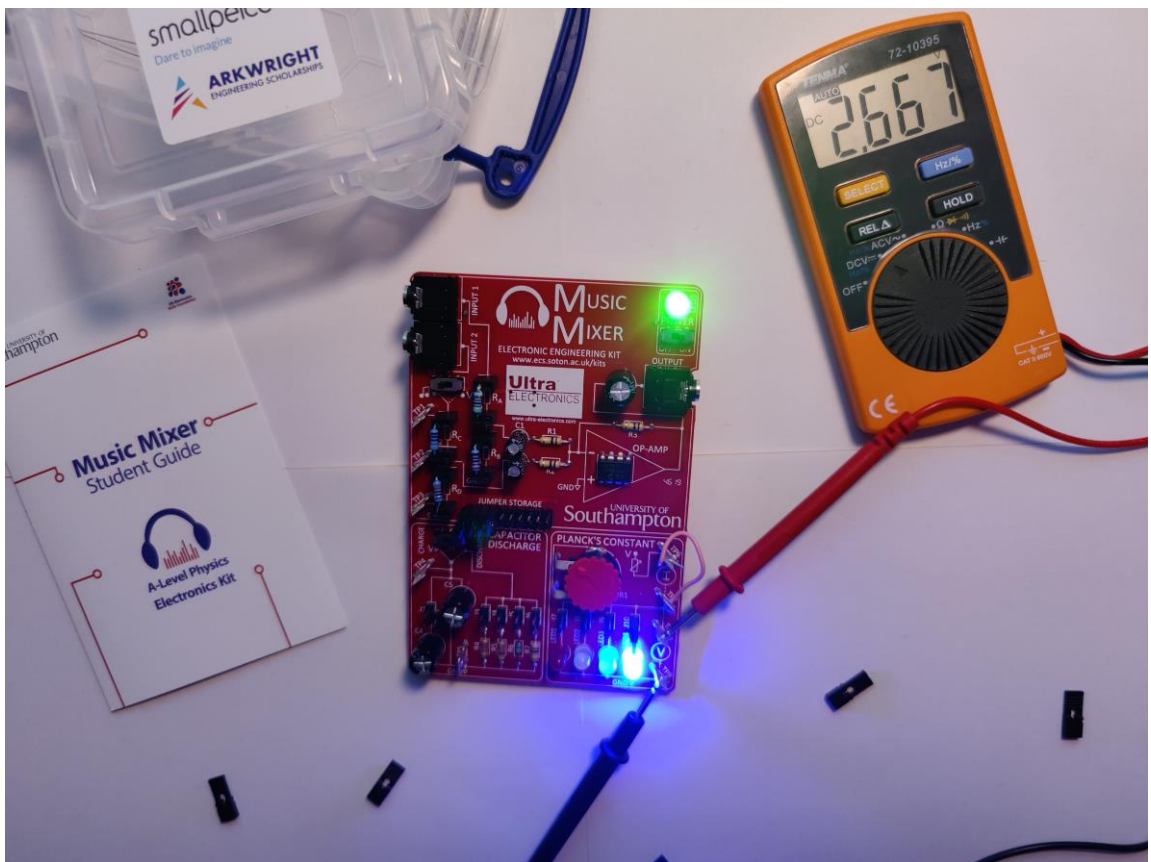
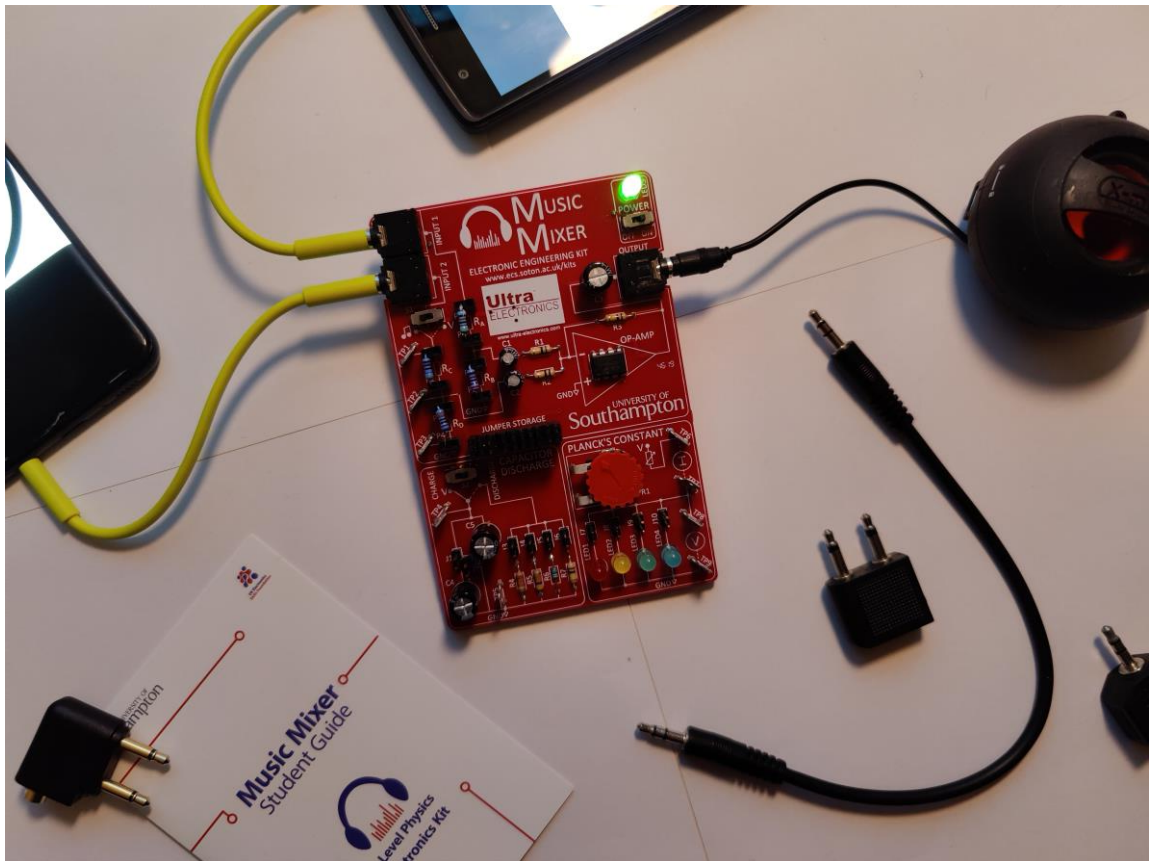
Replace  $R_A$  with an LDR such that there is an LDR placed on opposite sides of the potential divider for each input.



Repeat the process of covering and shining a light on the LDR's together. What is the difference between behaviour change when a single LDR was present versus now?

# G Stock Photo Highlights





# H Application Instructions

### Home Screen

This screen is scrollable

Supported kits

Links to find out more about:

- EE Kits
- UKESF
- Electronics Everywhere

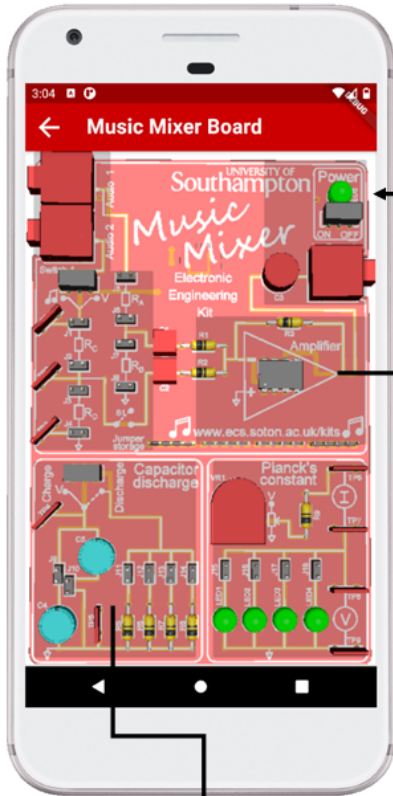
### Kit Screens

Interactive kit walkthrough

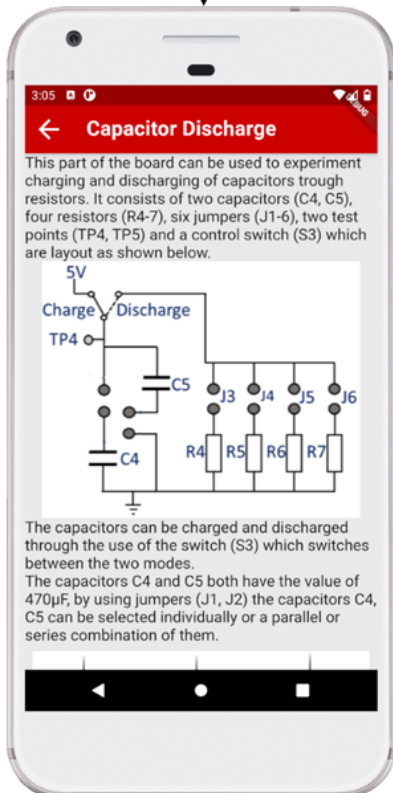
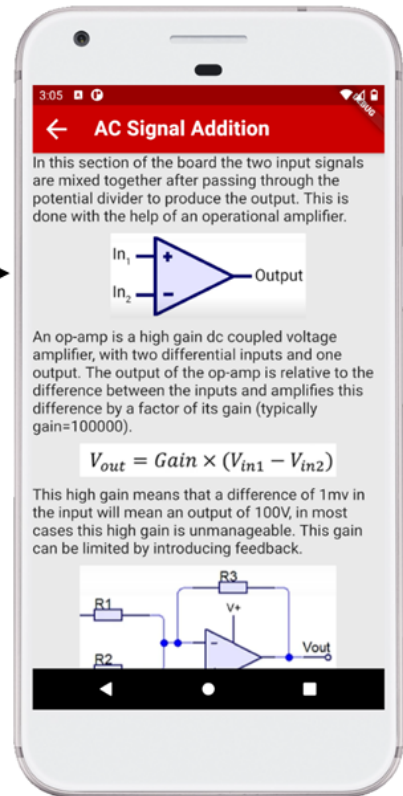
Brief information about the kits



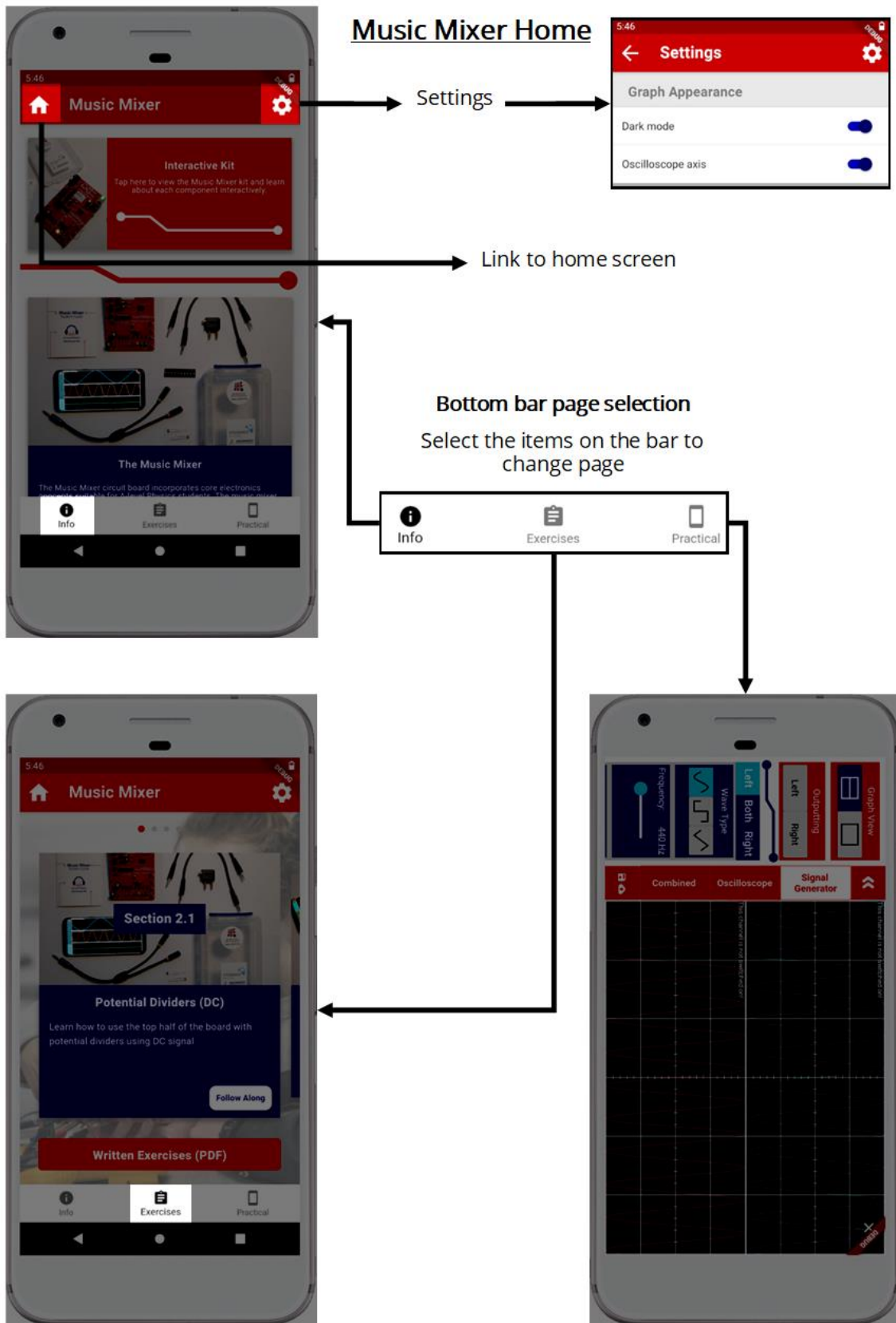
Interactive Kit



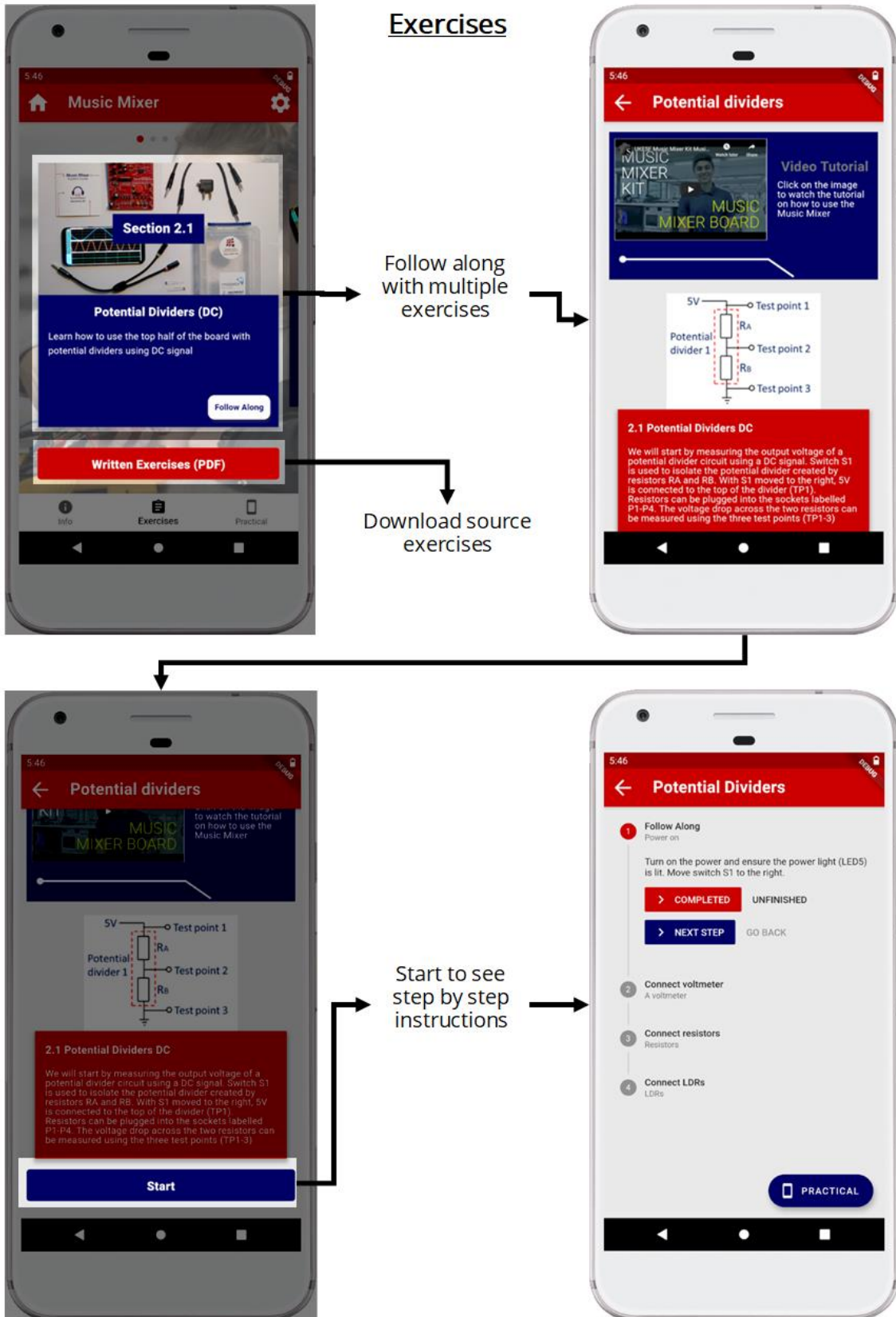
The shaded areas represent clickable sections of the board that you can open up



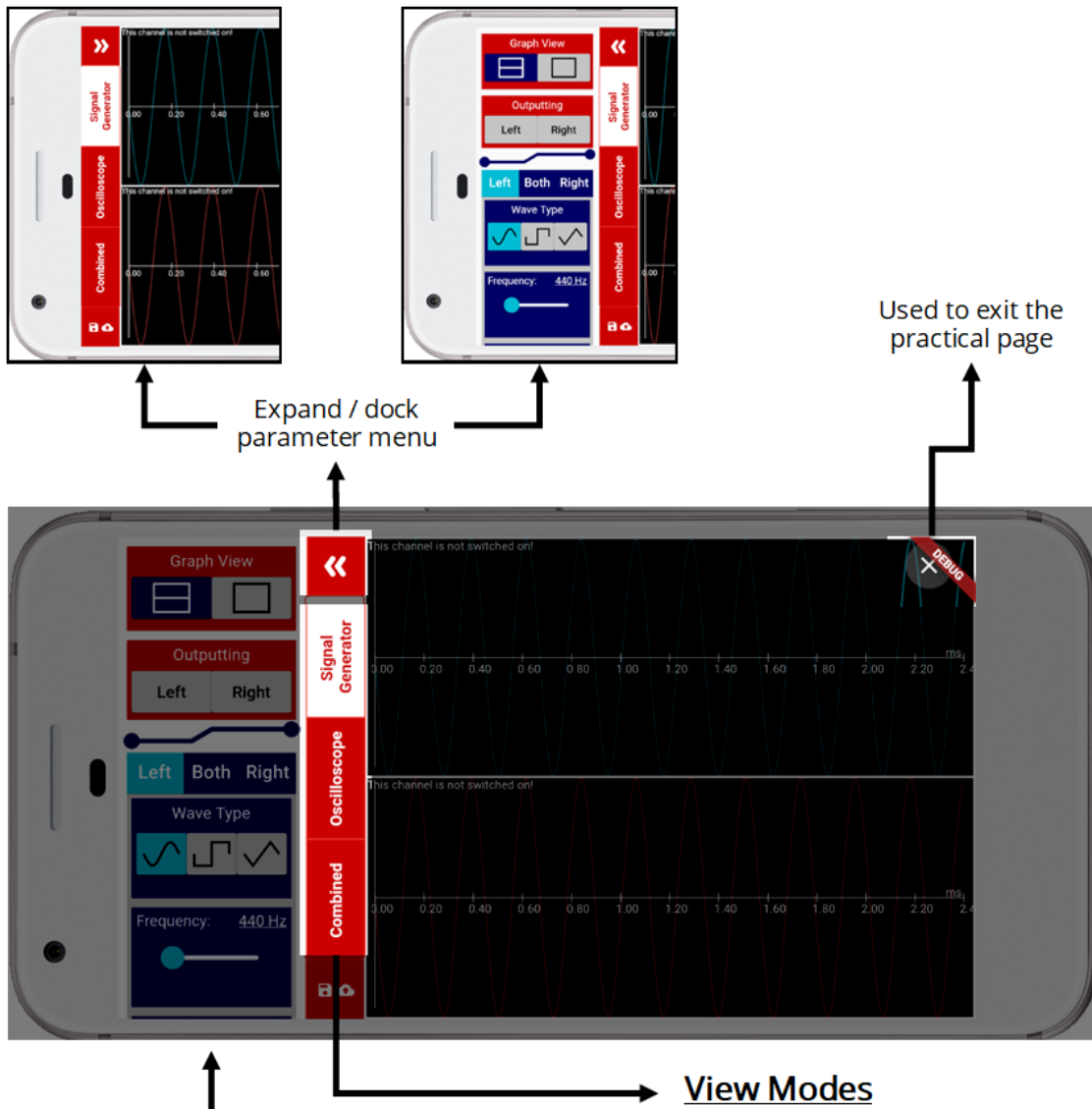
Information about that section of the board will be displayed. You can use these to learn about how the kit works



Exercises



### Practical Page Overview



Expand / dock parameter menu

Used to exit the practical page

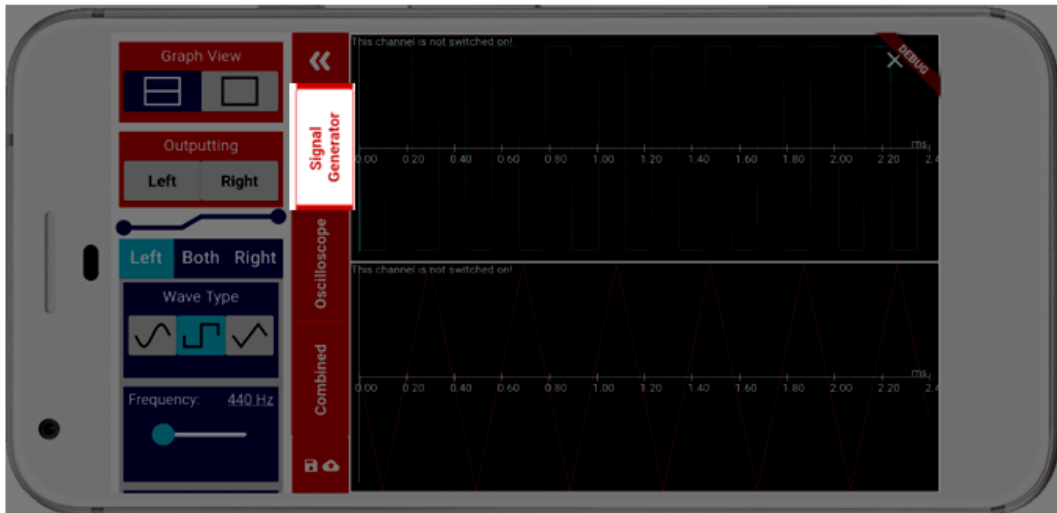
#### View Modes

The parameters menu changes based on the view mode that you have selected.

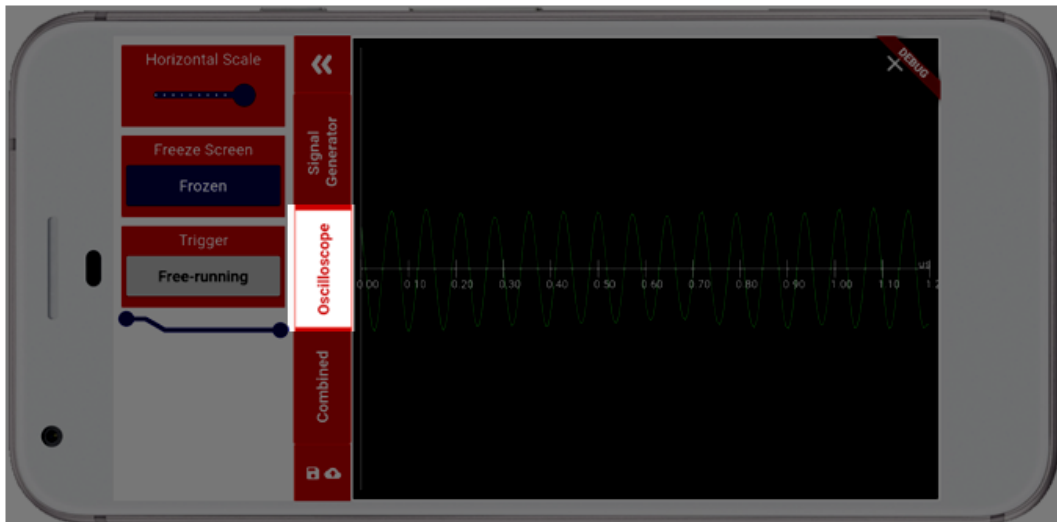
- There are 3 view modes:
- Signal Generator
  - Oscilloscope
  - Combined

These are demonstrated in the next page.

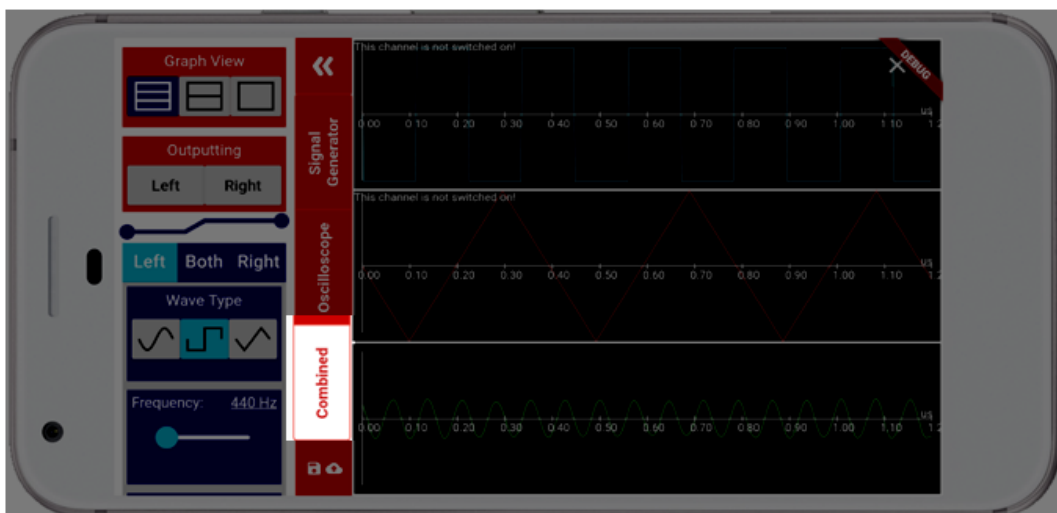
## View Modes



The **Signal Generator** can be used to generate signals to the input of the kit



The **Oscilloscope** can be used to visualise the output signal from the kit

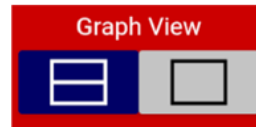


The **Combined** can be used to generate and visualise signals simultaneously

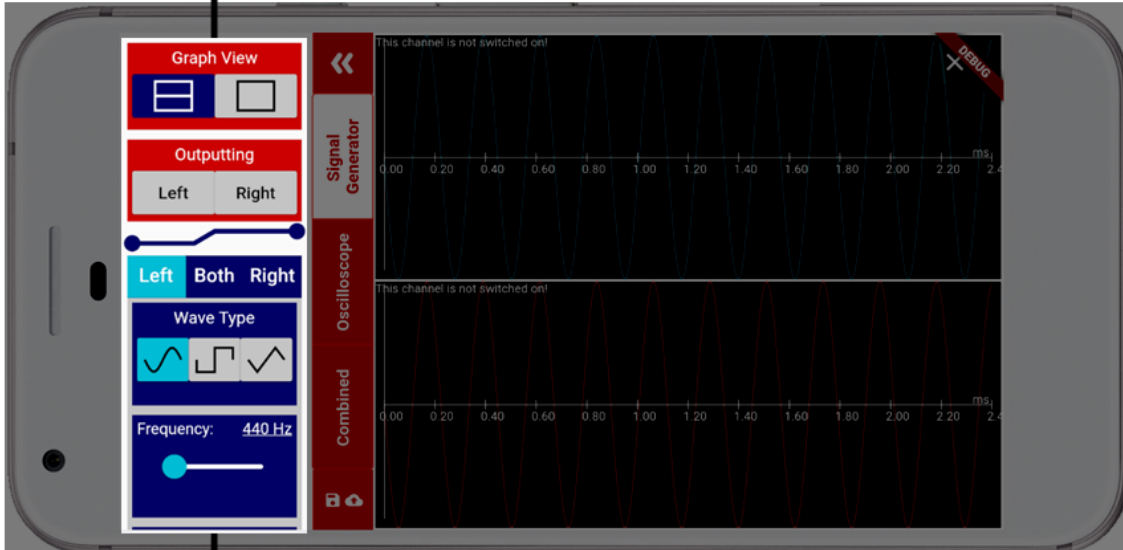
### Parameter Menu

Parameter menu (scrollable)

Changes how the graphs are split. These will be shown on the next page



Toggles which signals are outputting

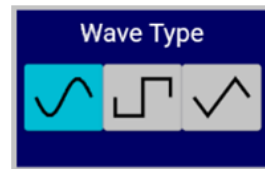


Parameter menu (scrolled down)

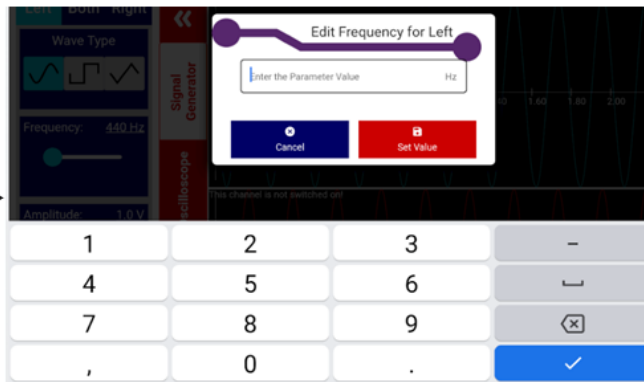
Tabbed selection of which signal is being changed



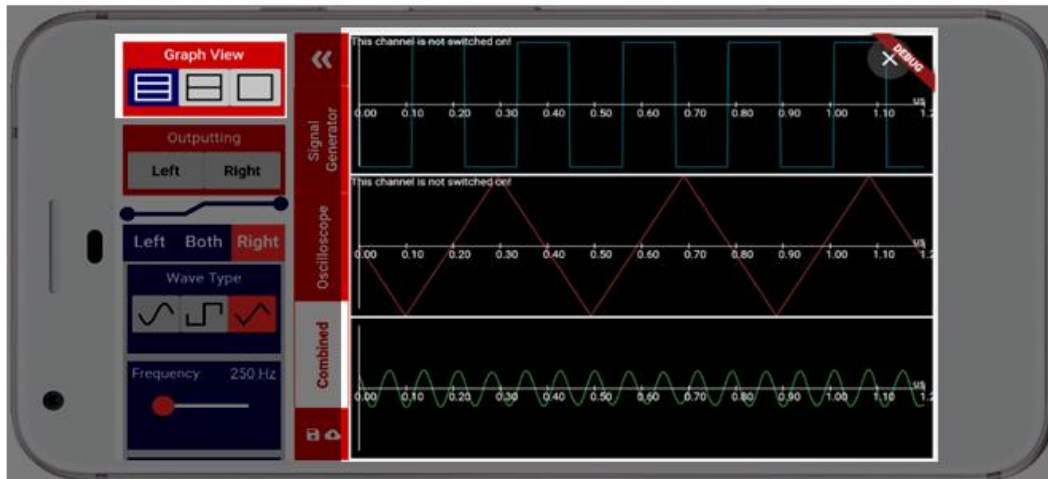
Wave type of selected signal



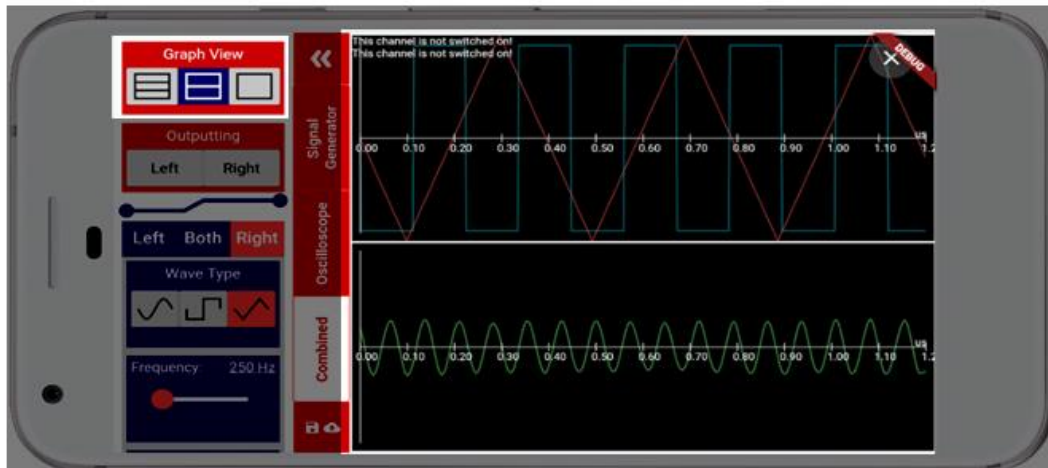
Manual parameter setting (click on underlined value)



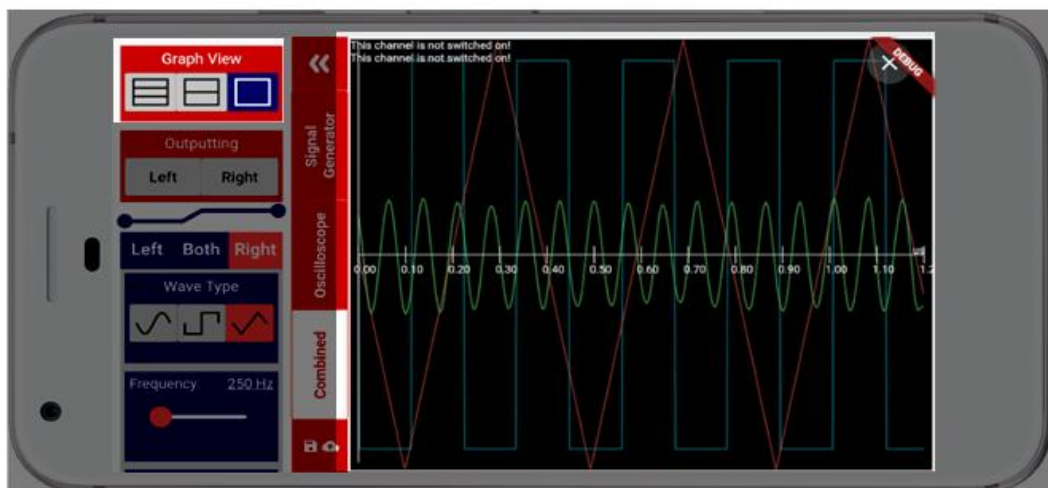
## Graph Layouts



This view has all signals displayed in their individual graphs

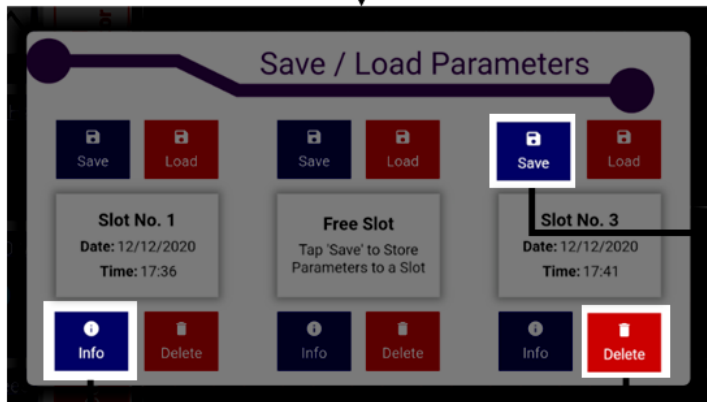
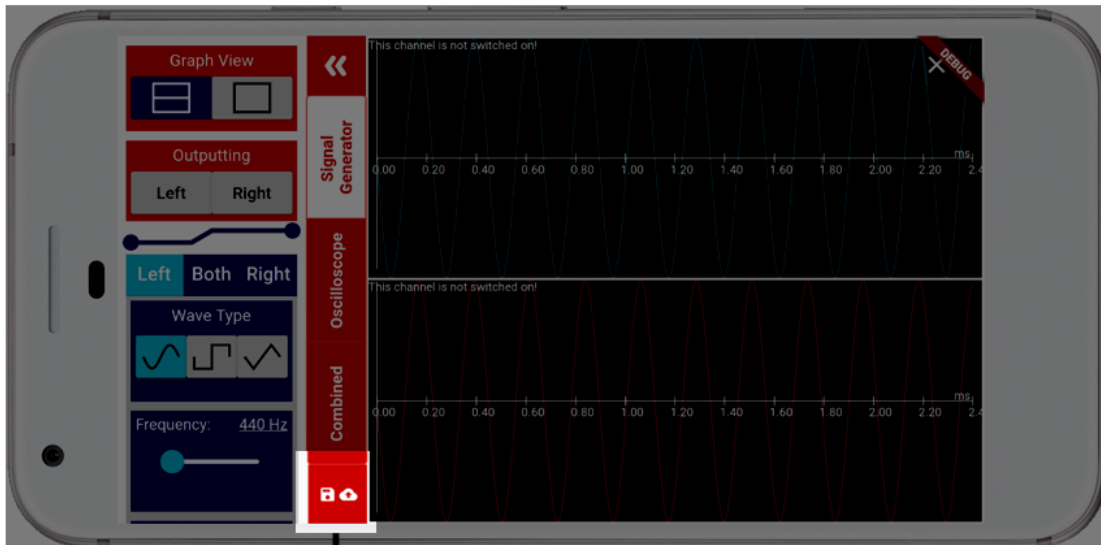


This view has the generated signals displayed in the same graph

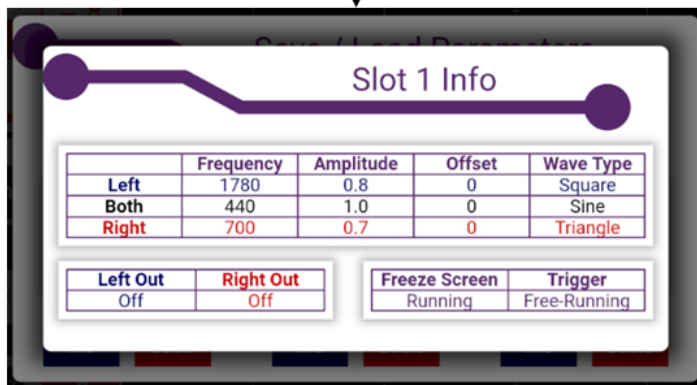


This view has all the signals displayed in the same graph

### Save / Load Slots



When you load from a slot, the save parameters will be set





# I Handover Document

## I.i. Introduction and Purpose of Document



Group Design Project 18: Electronics Everywhere Application Handover

### 1 Introduction and Purpose of Document

#### 1.1 Introduction

[Electronics Everywhere Application](#) is a project that aims to work alongside the UKESF / University of Southampton's outreach electronics kits to produce a cross platform (iOS and Android) application that contains the tools for students to be able to interact with the music mixer board without highly technical laboratory equipment. The UKESF's goal is to enable a more engaging first interaction between A-level physics students and electronics by facilitating the execution of A-level physics syllabus experiments that relate to electronics for both teachers and students. If the reader would like to become more informed about the origin of this project, or some of the process of development, they can find the groups academic report included in the project directory.

#### 1.2 Purpose

This document aims to give some insight into the current state of the project. It will discuss some of the suggestions the current team have for the software, hardware, and general direction of the project. These discussions will hopefully inform the reader about different limiting factors that were discovered during the project, including some of the hardware limited operation. These discussions are informed by surveys from A-level physics teachers and laboratory testing (the results of which can be found accompanying this document). The document also has information about the project's file structure. This handover report is a high-level brief information piece, and as such it is recommended that the reader also has access to the group report that contains in-depth information about the design and implementation of the application.



## I.ii. List of Features



### Group Design Project 18: Electronics Everywhere Application Handover

## 2 List of Features

Accompanied with this document is an “Application Feature Walkthrough” document. This outlines the way in which the features of the application works. As a reference, the following list has been composed for a brief overview of what has already been developed.

### 2.1 Available Pages

- Home Page
- Music Mixer
  - About
  - Interactive Image
  - Exercises
  - Settings
  - Practical
- Logic and Arithmetic
  - About
  - Interactive Image
  - Exercises

### 2.2 Available Features

#### 2.2.1 Home Page

- A header bar (at the top of the screen) that contains UKESF Logo and the name of the application
- Reference links to Electronics Everywhere, UKESF, and the University of Southampton’s Outreach Kits page
- Buttons to Music Mixer and Logic and Arithmetic Page

#### 2.2.2 Music Mixer Page

- Header bar (at the top of the screen) that contains a home button to go back to the home page, “Music Mixer” title and a settings link that changes preferences for the practical page
- A navigation bar (at the bottom of the screen) that allows the user to change the page between “About”, “Exercises” and “Practical”

##### 2.2.2.1 About

- Button to Music Mixer Interactive Image page
- A short description of the board

##### 2.2.2.2 Interactive Image

- An image of the board with buttons overlaid that give additional information about the physical operation of the board



## Group Design Project 18: Electronics Everywhere Application Handover

### 2.2.2.3 Exercises

- A list of four available experiments that can be followed for use with the application and the board
- A "Download PDF" button that allows the user to download the original source material

### 2.2.2.4 Settings

- A page with toggleable settings for the colours used (light / dark) and graph styling (standard / oscilloscope) for the Practical page

### 2.2.2.5 Practical

- Landscape orientation showing graphs that has the following modes of operation
  - Signal Generator
    - Produces independent (or identical) waves on the Left and Right audio output of the phone
      - Sine / Square / Triangle waves
      - 20 - 5000 Hz
      - 0 – 1 Amplitude (there is no distinct scale as this is hardware dependant)
      - 0-360° Relative Phase Offsets
    - Displays these waves on either a single, or two independent graphs
  - Oscilloscope
    - Records audio input through the microphone channel on the phone
    - Displays a single graph that displays the wave
    - Has toggleable triggering
    - Has toggleable freeze frame of the signal
  - Combined
    - Enables the previous operations of the Signal Generator and Oscilloscope on a single screen for viewing
    - Displays these signals on either three independent graphs, two graphs, with one containing the signal generation waves (and the other the oscilloscope), or three independent graphs

### 2.2.3 Logic and Arithmetic Page

- Header bar (at the top of the screen) that contains a home button to go back to the home page and "Logic and Arithmetic" title
- A navigation bar (at the bottom of the screen) that allows the user to change the page between "About" and "Exercises"



## Group Design Project 18: Electronics Everywhere Application Handover

---

### 2.2.3.1 About

- Button to Logic and Arithmetic Interactive Image page
- A short description of the board

### 2.2.3.2 Interactive Image

- An image of the board with buttons overlaid that give additional information about the physical operation of the board

### 2.2.3.3 Exercises

- A list of ten available exercises that can be followed for use with the application and the board
- A “Download PDF” button that allows the user to download the original source material



## I.iii. Known Bugs



### Group Design Project 18: Electronics Everywhere Application Handover

## 3 Known Bugs

### 3.1 Triggering on Negative Edge Zeros

This bug appears on the practical page and appears to be related to the limitations of using a phone to run the signal processing. It also is due to the method used to try and reduce processing power being used on the application, where the oscilloscope trace moves in steps, rather than displaying every point possible. This means that at certain frequencies and zooms, the graphs do not correctly line up between the oscilloscope and the signal generation. An example of this can be seen in Figure 1.

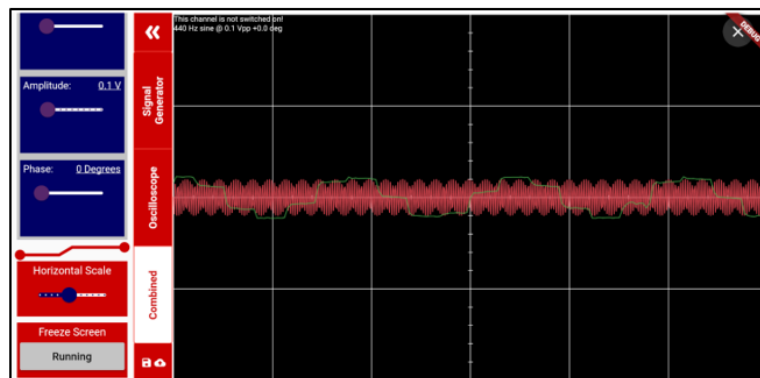


Figure 1: The display of the mismatch of the oscilloscope versus the signal generator display

Another issue is the fact that the zero point can jitter. This is just due to the method of detecting zeros, but it can mean that some of the time, at some scales, the graphs show a straight line from the end of the sampling, to the edge of the screen. This is, again, due to optimisations to not do processing on the entire range of samples, but rather on a small subset of samples that should be enough to be displayed. Since it is an imperfect system, however, this does not always work as intended. The effect can be seen in Figure 2.

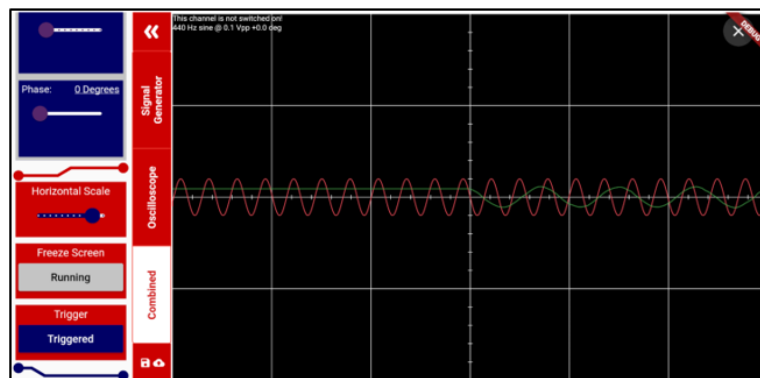


Figure 2: The display of the "jittery" line from the left side of the screen to the zero trigger



## Group Design Project 18: Electronics Everywhere Application Handover

### 3.2 Microphone Versus Unprocessed Input [Android]

This bug appears on Android only, since it is dependent on how the hardware manages the microphone input. `AudioFormat.MIC` or `AudioFormat.UNPROCESSED` react differently to the amplitude values. The issue is that for newer phones, unprocessed needs to be used, since it removes the amplitude varying after a few seconds of recording into the phone, but on older phones, using the unprocessed attribute causes a crash. As a result, there needs to be a check to see which version of android is running, or potentially on the hardware itself to see if the capability is available for unprocessed to be used. If, however, this check does not support unprocessed, mic will be used, and on those phones, there doesn't seem to be the amplitude modifying characteristic.



## I.iv. Software Information and Suggested Work



### Group Design Project 18: Electronics Everywhere Application Handover

## 4 Software Information and Suggested Work

There are some features that were investigated and discussed during the project that, due to the timing constraints, were not implemented. In this section, these features are discussed with information and mock-ups showing how the current team would have implemented them, their design considerations and any proposed action on the solution. This section assumes the reader is somewhat familiar with the application.

### 4.1 Slide Out Gesture for Settings Bar

#### 4.1.1 Idea / Feature

The settings bar is currently operated using the arrow button that expands and hides the settings bar with an animation. This is currently the only process to open the sidebar, so the suggestion is to add a gesture to slide out the settings bar that “follows” the gesture.

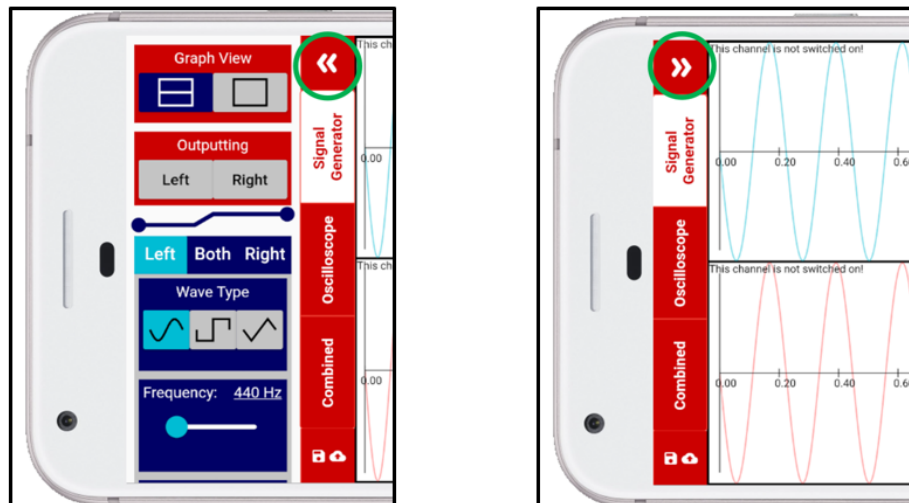


Figure 3: (Left) Settings bar in expanded mode (Right) Settings bar in hidden mode; Both have the control button shown

#### 4.1.2 Considerations

- A slide gesture can be more intuitive to users given the animation and settings bar would “follow” the slide out gesture
- The (potential) removal of the open/close button would make space for a dedicated back (or home) button as opposed to the current floating back button on the top right of the practical page
- Removing the visual indicator might remove the discoverability to the end user, e.g. having arrows on the side bar indicates you can open / close it, some visual addition would need to be considered in relation to this change
- This slide functionality may be intuitive to younger users, but to teachers and older users, it may not be as clear

## Group Design Project 18: Electronics Everywhere Application Handover

### 4.1.3 Proposed Action

The dedicated button should be removed and replaced with an “exit” button for the page (allowing the floating action button in the top right of the page be removed). In doing this, the developer should also add a visual indicator somewhere on the side bar to suggest that this sliding action can be performed, since otherwise there is no clear discoverability for the end user. Alongside this, the potential for an opening animation to start when the practical page is launched would also aid in demonstrating the action.

## 4.2 Links from the Experiment Instructions to the Practical Page

### 4.2.1 Idea / Feature

From the experiments page, whilst the user is following along to a lab, a button can be pressed to open the practical page with the appropriate settings. Leaving this page will return the use to the instructions allowing for easy interaction between the instructions and running the instructions.

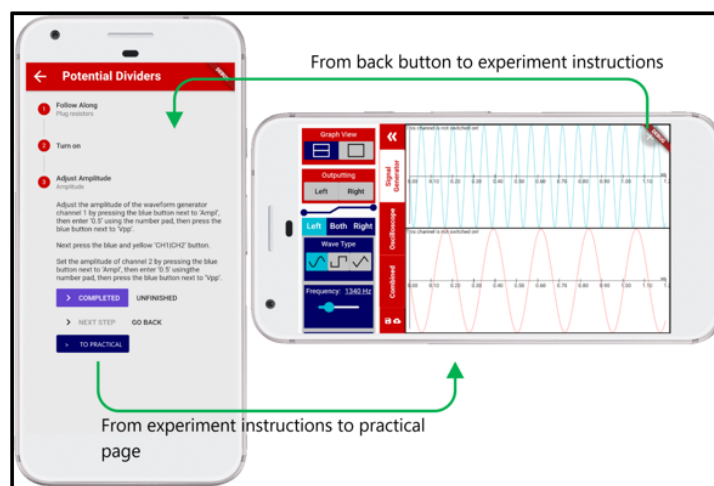


Figure 4: Mock-up of the transitions to and from the instructions and practical page

Interactive features such as text fields where a student may input their answers and results, possibly even getting feedback on their results. Examples may include:

- In 2.1, a table where the student may switch between the practical and exercises and input the information for the Voltages and Resistances.
- In 3, text fields for the student to input their values for the voltage observations. Possibly, graph plotting capabilities for plotting the results.
- For 4, a table so that the results may be inputting using the app as using a spreadsheet may be harder for some students on a phone.



## Group Design Project 18: Electronics Everywhere Application Handover

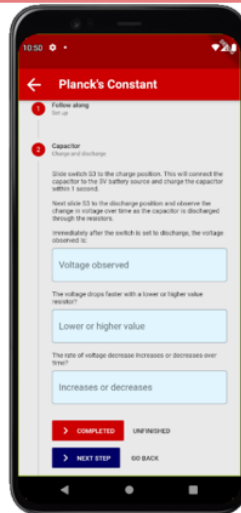


Figure 5: A demonstration of how interactive features have been added to the exercises

### 4.2.2 Considerations

- Users may want to use the app alone and thus read the lab instructions and have easy access to the practical page at the same time
- The back-button functionality on the practical page and as such should indicate a “back” icon, not an exit one
- The settings for the experiments should be able to be set when opening the practical page from the instructions (i.e. the instructions reference a 440 Hz sine wave, and this is a pre-set entered when opening the page)
- Automatically setting the values might take away from the engagement students receive when running the instructions themselves (changing the sliders, editing the signals) and might prevent the fact students might explore with other settings (such as using settings other than the suggested from the lab)

### 4.2.3 Proposed Action

To implement a floating action button on the instruction pages (where it is relevant to enter the practical page). Below shows an example of a floating action button that has been implemented with the blue UKESF branding. This button will have custom functionality to set the starting variables of the page (through shared preferences) and so when opening the page, it can be set to the relevant settings of the instructions.



Figure 6: Currently placeholder button for this operation

## 4.3 Changes to the Interactive Image to become a 3-Dimensional CAD Model

### 4.3.1 Idea / Feature

The Interactive Image could be further improved by replacing the 2-Dimensional Image with a 3-Dimensional Model that would allow the user to visualize all aspects of the Kits using finger gestures such as 'pinch to zoom' and/or 'swipe to rotate'. The Model could also automatically zoom and move to a specific element whenever a certain section is tapped.

### 4.3.2 Considerations

- A simpler CAD Model could be designed with 'less detail' than the Kit to have an implementation that is less technically difficult and less computationally expensive (when referring to performance).
- It would generally be wise to constantly have the 3D model in 'free-view mode' so that the user is able to freely 'explore' the board in whatever way they enjoy.
- A tutorial in the form of a mini popup should be displayed on the first launch of the Interactive Image for the user to know the instructions & various finger gestures that could be used to avoid confusion.
- Whenever an Area of the Board is tapped (e.g., Operation Amplifier, Planck's Constant, etc.) the Model will automatically rotate & zoom in on that element. Additionally, a Popup will be displayed indicating a small amount of key information with the option to 'Read More' as described in the animations of the interactive board section.

### 4.3.3 Proposed Action

Initially, a CAD Model will have to be designed using a 3D computer graphics application such as Autodesk Maya. This can easily be rendered in Flutter using external packages such as `model_viewer` which allows for rendering interactive 3D models in the glTF and GLB formats.



## Group Design Project 18: Electronics Everywhere Application Handover

### 4.4 Tutorial of the Practical Page Embedded in the Application

#### 4.4.1 Idea / Feature

The practical page does not currently have a tutorial baked into the app. This is not a major drawback as the app is relatively simple, but an interactive tutorial might help some discover all the features available.



Figure 7: Mock-up of practical page tutorial / walkthrough

#### 4.4.2 Considerations

- This tutorial would need the ability to be switched on or off
- This tutorial would start on the first run of the app
- The tutorial video can be published so that students can discover all the features through the video

#### 4.4.3 Proposed Action

Using the same framework as the pop ups on the interactive image page, set a sequence of pop ups and displays that explain what the buttons do in which positions. Have this run once, then set a variable in "sharedPreferences" that means that it will not run again.



## Group Design Project 18: Electronics Everywhere Application Handover

### 4.5 Accessibility Text Size

#### 4.5.1 Idea / Feature

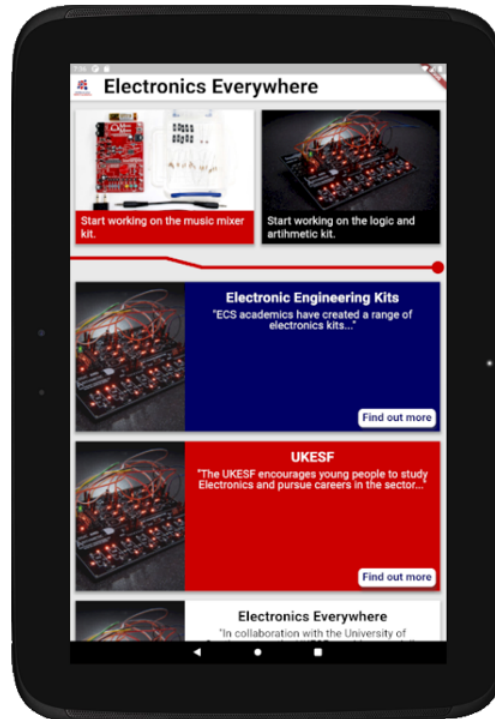


Figure 8: Tablet scaling with the current implementation in Flutter

On most devices, there is an accessibility setting that will change the text size globally on the phone. This global setting can sometimes cause formatting issues in an app with tight constraints. In addition to this, depending on the device, the text may need appropriate scaling in order to improve readability and aesthetic. With fixed text sizes, text may be less readable on larger phones, so text scaling should be used to improve this. This is also a possibility for containers and especially – the app bar and bottom navigation bar which have fixed sizes.



Figure 9: (Left) Application with accessibility settings default (Right) Accessibility options at maximum size



## Group Design Project 18: Electronics Everywhere Application Handover

### 4.5.2 Considerations

- Trying to meet the accessibility legal requirements for a web application, (Web Content Accessibility Guidelines) WCAG 2.1, as best as possible.
- This makes the application more accessible and inclusive to more people
- The largest phone sizes should not drastically scale the text and UI
- The largest phone sizes to name a few:
  - 3088x1440 (Samsung Galaxy Note 20 Ultra)
  - 2778x1284 (iPhone 12 Pro Max)
  - 2208x1768 (Samsung Galaxy Z Fold 2)

### 4.5.3 Proposed Action

Most of the fixed text sizes are set in the view/screens/theme.dart file, and the text scales depending on the width of the phone screen. The text scale factor for Text widgets has been set to the screen width / 480 which is what seems to fit best with the text sizes set, however this may be changed.

Changing the fixed size containers to dynamically change size would improve the app experience for students who will likely have a variety of different phone sizes. This includes the app bar and bottom navigation bar, that have a preferred size that should not be fixed since it shows also scale if the title scales.





## Group Design Project 18: Electronics Everywhere Application Handover

### 4.6 Updating Music Mixer Exercise Instructions

#### 4.6.1 Idea / Feature

The steps in the Music Mixer Exercises, especially for the section 2.1, could be updated with images, instructions and content using the application instead of the third-party app and/or matched with update lab notes if the decision is made to update the existing lab notes. These changes will need approval from both the UoS ECS department and the UKESF.

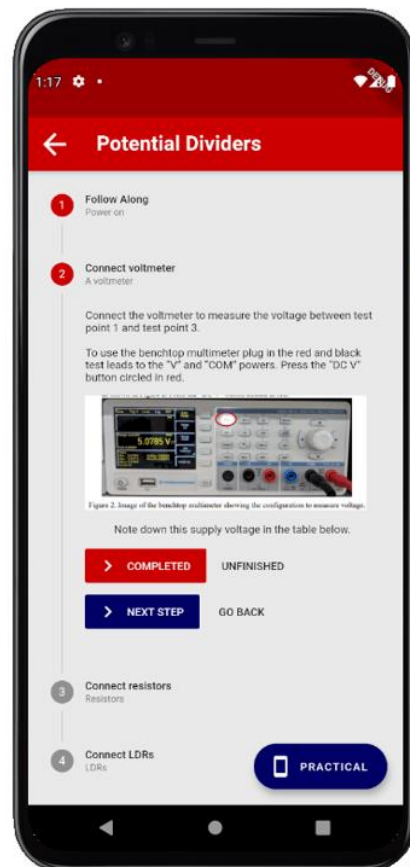


Figure 10: Exercise step for lab 2.1: Potential Dividers (DC) showing image of oscilloscope from the Training Handbook

#### 4.6.2 Considerations

- Depends on whether the decision is made to change the lab note instructions with use with the app
- Whether the image of the oscilloscope is still helpful with most classrooms probably not having access to them
- Whether the information on how to use the third-party software and application is still necessary for the lab notes or should be replaced with the companion app on release



## Group Design Project 18: Electronics Everywhere Application Handover

### 4.6.3 Proposed Action

If the decision is made to add to, or update, the existing lab notes then the images for the lab steps and other content can be changed easily in their corresponding class in view/screens/labs/.

An alternative version of the source notes has been modified to include instructions that use this application, and a small snippet of this can be seen in Figure 11. The changes made are focused around 2.2, with also an addition to 2.3. This will be included in the handover files.

Turn the music mixer board off. Plug the “airplane” splitter into the inputs of the music mixer board. Connect the “airplane” splitter to the headphones part of the splitter (usually green). Connect the output of the music mixer board to the microphone part of the aux splitter (usually shown in pink). Connect the male end of the splitter into a phone’s auxiliary port. The setup should look something like this:



Figure 11: A small snippet of the proposed modified instructions

## 4.7 Additional Dynamic Scaling for Proportions

### 4.7.1 Idea / Feature

There is a wide range of devices that could use this application with a variety of different screen sizes. The app does currently have dynamic scaling based on the devices screen size in some areas (such as the sidebar on the practical page). The current implementation has features being proportional to the screen size only. The proposition is that there are ranges in which this affect takes place, as for tablet screen sizes, this might lead to overly large settings bar.

## Group Design Project 18: Electronics Everywhere Application Handover



Figure 12: (Left) Application being used on a phone (Right) Application being used on a 10" screen tablet

### 4.7.2 Considerations

- The potential to have proportional scaling that works within some set minimum and maximum values e.g. the settings bar might not always be a set proportion of the screen width, because for very large screens, this might look disproportional
- The typical use case is phones, but teachers regularly have tablets used for educational purposes like this
- Providing these options also open more options to teachers (potentially using an interactive whiteboard to display using an emulator)
- This would need to be tested on a variety of devices with various screen sizes and ratios to fine tune the proportions and ranges

### 4.7.3 Proposed Action

These scales can be quickly implemented through a "textScaleFactor" attribute that can be changed depending on screen size and aspect ratio.

## 4.8 Cross-Platform Support for Web Browsers

### 4.8.1 Idea / Feature

Flutter has support to convert the current application to work with web browsers and this would be another good addition to help remove the barriers of use for the application.

### 4.8.2 Considerations

- Permissions for web browsers may work extremely differently to the way that mobile permissions work
- The way the audio input and output will have to be connected might be very different for a desktop compared to a laptop, essentially increases the amount of variability of operation based on hardware
- The content would have to work on a mobile centric platform (working like bootstrap for websites) so that the content would adapt
- Trying to meet the accessibility legal requirements for a web application, (Web Content Accessibility Guidelines) WCAG 2.1, as best as possible.





## Group Design Project 18: Electronics Everywhere Application Handover

### 4.8.3 Proposed Action

Enable the use of the cross platform on web browsers but ensure that there is enough warning about the difference in support. Most of the application will work the same way, except and if the previous points have already been addressed, the content should work in a manageable way.

---

## 4.9 Logic and Arithmetic Support

### 4.9.1 Idea / Feature

The logic and arithmetic board's inclusion in the app now have the interactive board walkthrough and the ten exercise instructions from the lab notes. Further support would include a separate piece of hardware that interfaces between the board and the phone.

### 4.9.2 Considerations

- The hardware required to interface with the logic and analyser board will be more extensive than interfacing with the music mixer board
- The board could also be redesigned to include an interface section to use with the app
- The signals that would need to be sent between the board and phone would be digital
- The protocol for communicating between the phone and the hardware would probably best use Bluetooth as opposed to the headphone jack, down to the nature of the signals
- There would be a large cost (time and resources) to design and produce such hardware, and the gain of interfacing this with the phone to use as a logic analyser might not be worth it

### 4.9.3 Proposed Action

Develop a new piece of hardware that joins the requirements of both the Logic and Arithmetic board and the Music Mixer board. Essentially a USB c or lightning port that can report voltage values of multiple inputs. This can be used to show more traces on the oscilloscope, or to report different digital signals for the Logic and Arithmetic board.

---

## 4.10 Pinch to Zoom for the Oscilloscope / Signal Generator

### 4.10.1 Idea / Feature

The current method of changing the scale of the oscilloscope (and all the graphs on the combined) is by changing the horizontal scale slider. A pinch gesture could be used in addition to this to change this scale.



## Group Design Project 18: Electronics Everywhere Application Handover

### 4.10.2 Considerations

- It might be more intuitive to some to pinch in order to zoom into a graph
- Both the slider and the gesture could be in place, giving the option for both
- The gesture might cause issues with others, in reference to distinguishing multiple different gestures e.g. sliding open the settings bar

### 4.10.3 Proposed Actions

Add the pinch zoom on the signal generation and oscilloscope page but ensure that it is only a horizontal scale adjustment. It should also still change to the automatic scale when changing the signal generation page but allow manual scale if nothing has changed. On the oscilloscope page, the pinch should just change the scale for the oscilloscope and be limited by the sliders.

## 4.11 Graph Colouring and Overlay Style

### 4.11.1 Idea / Feature

The current method the graph displays the waves on top of the axes and the axes labels. This means that the graphs can hide the scale information if the scales are not set correctly to be able to see the details of the graph. The current implementation can be seen in Figure 13.

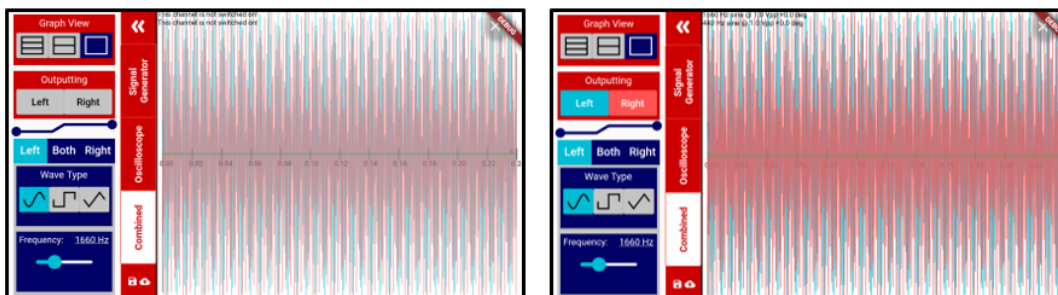


Figure 13: A demonstration of the current implementation of opacity to show the axes with (Left) the signal outputs off (Right) the signal outputs on

### 4.11.2 Considerations

- The graphs should not be used at this scale, since it would be impossible to see the detail anyway, the recommendation would be to zoom in
- This is only something that happens on the combined screen, since on the signal generation page, the graphs scales are determined from the signals themselves, and so will never reach this critical point to cover the axes
- Currently the graphs are displayed with some opacity, and so the axes can be seen
- This opacity is reduced (the waves become opaquer) when the signals are outputting, and this causes the axes to become more hidden at incorrect scales



## Group Design Project 18: Electronics Everywhere Application Handover

### 4.11.3 Proposed Action

Find the optimal colours and opacities to show the difference between being on and off, and to ensure that at any scale the graphs can be seen. It might be a case of ensuring that if the scale and signal generated signals are extremely mismatched, that the axes are drawn above the waves, and otherwise they are drawn below.

## 4.12 Multi-theming Options

### 4.12.1 Idea / Feature

A lot of high-end mobile applications have different theming options, for example dark/light modes and high contrast mode. The modes are mainly for aesthetic reasons however the high contrast mode makes the only colours white and black for easier reading.

### 4.12.2 Considerations

- Different modes are probably not necessary
- Having different modes may have a negative impact on the UKESF branding theming
- A dark theme colour branding might need to be designed

### 4.12.3 Proposed Action

Flutter makes it very easy to have a dark mode where the colour may be set and a simple switch on a global settings page may be added to toggle between modes like the one for the music mixer page. There is also a theme dart file with all the colours used in the application making it easy for more colours and themes to be added using the ThemeData class. It may also be useful to investigate the accessibility guidelines for contrast modes for the UK.

## 4.13 Triggering Based on Movable Trigger Point

### 4.13.1 Idea / Feature

Currently, the triggering is only on 0. This feature would allow the user to move the trigger point in order to correctly trigger on more interesting waveforms, such as superposed waves of different frequencies.

### 4.13.2 Considerations

- Helps to generalise the oscilloscope to more uses beyond being limited to the application
- This might confuse students as it might not be clear what a “trigger level” does in this context
- The use case might be extremely small, and therefore might not be worth complicating the GUI
- If the trigger is implemented, how it works on the oscilloscope page needs to be repeated on the combined page where there is also the signal generation graphs



## Group Design Project 18: Electronics Everywhere Application Handover

### 4.13.3 Proposed Action

I believe the best solution would be to include this movable trigger option but have a popup that appears the first time the user moves the slider that explains what it does (or potentially a question mark icon for this pop up). This would allow the functionality, without a student understanding. I also would suggest that the signal generator should move compared to this trigger level rather than add a manual offset. This reduces the complexity for the end user. The triggering for the signal generator would only work under two considerations, however. The first is that the user is using the combined mode for the reason of outputting from one phone, and back into the same phone. The other consideration is that the two waves are trivial enough to be able to add together to find at least one trigger point. Once the initial trigger point is found, this can be used for the offset.

## 4.14 Support for Microphone Inputs to Phone Directly from the Board

### 4.14.1 Idea / Feature

The board currently does not have an output that maps to an input on the phone. As a result, a potential suggestion is to add these functionalities to the output of the board so that a no in-line attenuation needs to be added, since it is already on the board. The specific hardware requirements are discussed later in 5.2.1.

### 4.14.2 Considerations

- The audio output may only want to be used for audio playback and not microphone
- The microphone lines require an impedance and also a resistance difference to GND to be detected
- The additional cost on the board may, or may not outweigh the cost of adding an in-line attenuator that is required for easy detection of the microphone on a phone
- Adding a specific output for microphone would remove the ability to output and take an input from the same phone (as the left and right channel would be split incorrectly into the splitter for microphone).
- These suggestions are also dictated by the number of phones typically interfacing with a single board. If the most common use case is 1 phone, an “all in one” port would be beneficial, but if 3 phones were typically being used with one board, having 3 separate ports (similar to the current layout) would be more beneficial
- By having one port on the board, would mean extra adapters if multiple phones were to connect with a signal board, whereas if there were multiple ports on the board (similar to the current layout), more adapters would be needed for a single phone to interface with the board

## Group Design Project 18: Electronics Everywhere Application Handover

## 4.14.3 Proposed Action

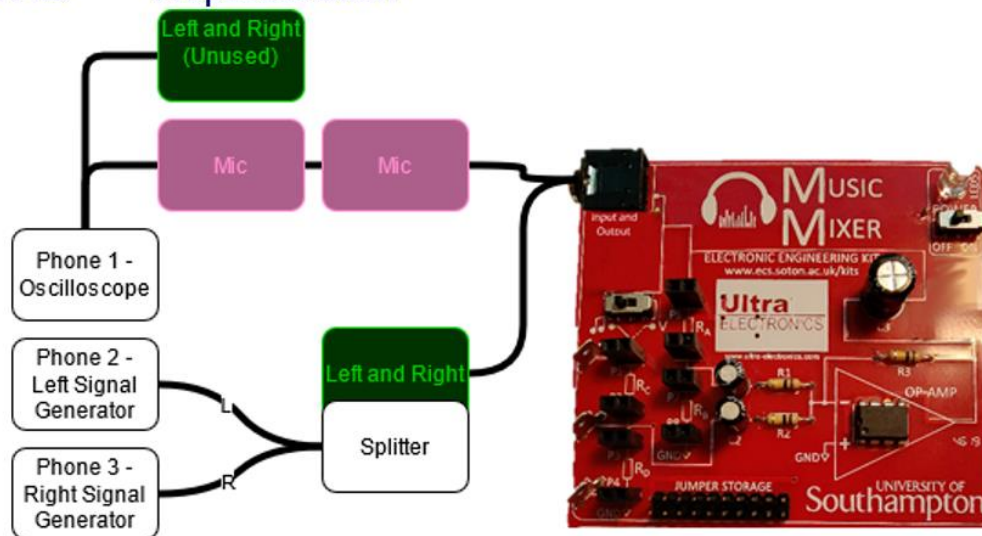


Figure 14: Mock-up of the board redesign with TRRS connection to 3 phones with splitters

One option is to replace the inputs and outputs on the board to be a single 4-Segment Plug (TRRS connector - <https://uk.rs-online.com/web/p/jack-plugs-sockets/8051665/>) into the board. In doing so, only a single connection cable would be required from the phone to the board (a simple TRRS cable). This is assuming that the output for the microphone conforms to the requirements stated in 5.2.1 for microphone detection. There are two benefits from this, a reduced cost in ports placed on the board, and a reduced cost in the number of adapters required to interface. The downside would be the fact that splitters would no longer be as easily connected to the board, however, could still be done with the recommended splitters, and can still be used with multiple phones at a single time. A mock-up is shown in Figure 14. The true benefit of this solution is shown in Figure 15, where a single phone interfaces with the board.



Figure 15: Mock-up of the board redesign with TRRS connection to a single phone that is used as both oscilloscope and signal generator



### Group Design Project 18: Electronics Everywhere Application Handover

Another other option is to add another output to the board, exactly the same as the current output (with left and right connected exactly the same way) apart from the fact that there would be an impedance large enough on this line that it would be detected as a microphone if the op-amp was on or not.

The recommendation is to add the single port, whilst providing the splitters that are suggested in order to be able to continue to connect multiple phones to the same device.



## I.v. Hardware Information and Suggested Work

### Group Design Project 18: Electronics Everywhere Application Handover



## 5 Hardware Information and Suggested Work

This section discusses the discoveries made about the required hardware with the application. It will also discuss the behaviour that has been verified in a laboratory with different interface hardware.

### 5.1 Required and Compatible Hardware

#### 5.1.1 Required splitter

To interface between the music mixer kit and the application on the phone, one further piece of hardware is required. To connect the output of the music mixer board to the microphone part of the headphone jack of a phone, a splitter is required. If a standard 3 ring auxiliary cable were connected directly from the output of the music mixer board to the phone, the output from the board would be connected to the left/right output of the phone, which would mean that the phone would not be able to read in any data. To remedy this, using a single phone, it is possible to output to, and take an input from, the music mixer board. The mappings of the splitter can be seen in Figure 16. An example of how to connect this cable with the board is shown in Figure 17.

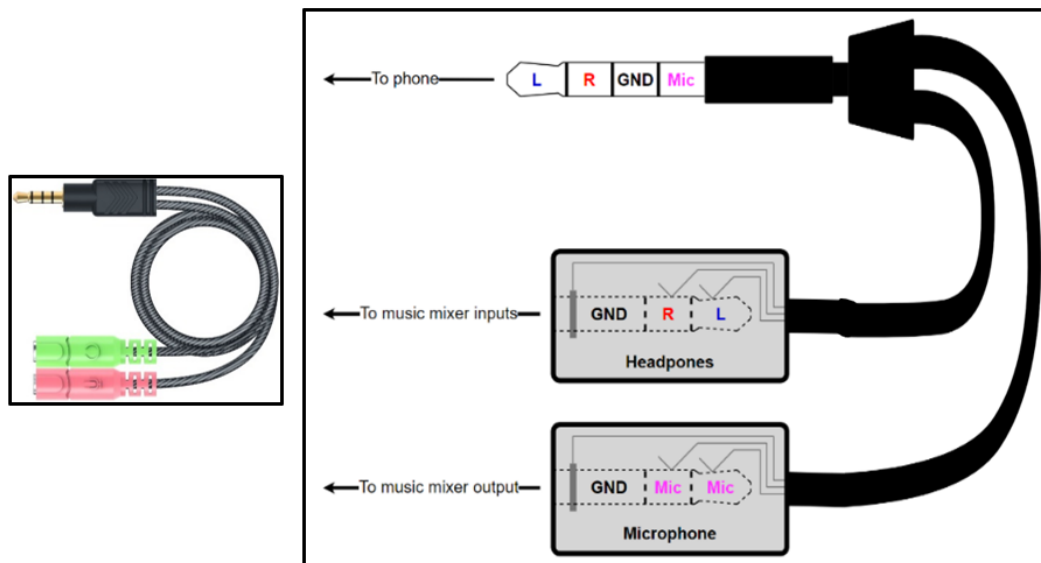


Figure 16: (Left) Example of the type of splitter required (Right) Splitter diagram with pin mapping

Using only these adapters with two android phones (a Samsung S8 and a OnePlus One) it was found that there is a difference in absolute amplitude from the output port. This means that when taking the amplitude measurements, there can be not absolute measurement on the oscilloscope, since both phones have a difference capacity for outputting, but also for the maximum measurement that they can receive into the phone. The results of the amplitude testing are included in the project archive.

## Group Design Project 18: Electronics Everywhere Application Handover



Figure 17: Demonstration of how to plug in a single phone using the connector

### 5.1.2 Other hardware adapters

Part of the hardware testing involved using a variety of different configurations of adapters. This was done so we had a complete overview of the system in different use cases. The following sections will describe our findings about using certain adapters.

#### 5.1.2.1 USB to auxiliary adapter

The USB to auxiliary adapter behaved in the same manner as without, with the difference being that the amplitude of the signal output was measurably lower than without the adapter. *This is potentially due to the cheap adapter used, but it is unlikely.* Another key difference is that the USB to auxiliary adapter would only output to the port if there were correct resistance and impedance on the lines (much like the same way the microphone only detects a line in if the resistances and the impedances are correct).

#### 5.1.2.2 Bluetooth speakers

The Bluetooth speakers outputted the audio in the same way that it did on any physically connected device, however at a slightly different amplitude. This supports the suggestion that there should not be absolute measurements on the amplitude scale.

#### 5.1.2.3 In-line attenuator

IOS devices seemed to be fussier than Android devices when it came to the detection of microphone input. The limitations on microphone detection are discussed in 5.2.1.

## 5.2 Hardware Limitations

This section will discuss the limitations of using phones, the limitations of going cross platform solution, and the limitations of the hardware of the auxiliary port on a phone in connection to use with the music mixer board. These limitations were found during testing of the application on multiple phones in a laboratory with oscilloscopes and signal generators. All the recorded results can be found in the project archive.





## Group Design Project 18: Electronics Everywhere Application Handover

### 5.2.1 Microphone Detection and Processing

In order to detect the hardware that is connected to it, Android phones have specific hardware requirements to identify the use of the lines for the auxiliary port. This is done so that the phone knows how to handle the connection (i.e. is it an input or an output).

The specification for detecting a microphone input to the phone is to have an 100 $\Omega$  resistance across the line to the GND pin. The impedance also should be above 5k $\Omega$  to be detected as a line in. [<https://source.android.com/devices/accessories/headset/jack-headset-spec>]

The board has an op-amp connected to the output, and as a result means that when turned on, it is not detected as a line in when plugged in to the phone. A simple workaround for this is to switch off the board for about 5 seconds, turning the op-amp off and causing the output to have high impedance.

Testing on several android devices, once the phone has detected the line in, it does not change this unless the port is unplugged. The issue appears similarly on iOS devices, except that iOS does seem to ensure that this line impedance is always correct. The solution to this has been mentioned above with the in-line attenuator, which puts the correct impedances and resistances on the lines in order to be detected as a microphone.

Another solution to this would be to redesign the output of the board in order to have these resistances and impedances already on the line so that no additional cables would be needed. The only issue would be that the current boards would need the additional hardware to work with the application.

### 5.2.2 Signal Generation

#### 5.2.2.1 Frequency

Due to the hardware limitations of the headphone jack and related circuitry, the maximum frequency at which we can send data is 44.1kHz. This means that to generate a sine wave of frequency of 4.41kHz, there are only 10 discrete points per wavelength. We chose a maximum limit of the frequency generated by the app to 5kHz, which was chosen because it was a compromise of frequency and resolution. Since the application produces a sine, square and triangle wave, the 5kHz is a good maximum since it allows 8 points per period, which should be enough to be differentiable between the wave types. Based on testing with a Samsung S8 the frequency had an average accuracy error of 0.66%. Testing on an iPhone XS there was an inaccuracy of 0.14%.

#### 5.2.2.2 Amplitude

Different devices have different hardware components and processing applied to signals. This means that the absolute magnitude of the generated output waves varies. When running tests on the Samsung S8, with various dongles, and the OnePlus One, the amplitude of the phones varied greatly at each incremental volume. A list of key results



Group Design Project 18: Electronics Everywhere Application Handover

from the different phones models are shown in Table 1. This supports the idea that there cannot be a labelled vertical axis (other than a relative one) as the amplitudes vary based on the hardware used.

Table 1: Results of amplitude results from testing on different phones

Phone Model	Volume: 0	Volume: 0	Volume: HALF	Volume: HALF	Volume: MAX	Volume: MAX	Volume: MAX
	Amplitude: 0	Amplitude: 1	Amplitude: 0.4	Amplitude: 1	Amplitude: 0	Amplitude: 0.4	Amplitude: 1
Samsung S8	20.8	20.8	90	208	20.8	912	2240
OnePlus One	20.4	20.4	121	266	20.4	1120	2760
iPhone XS	25	36	90	178	42	1120	2720

5.2.2.3 Phase

There is a phase inaccuracy of 0.49% on iPhone, and 0.85% on Samsung S8.

5.2.2.4 Wave Type

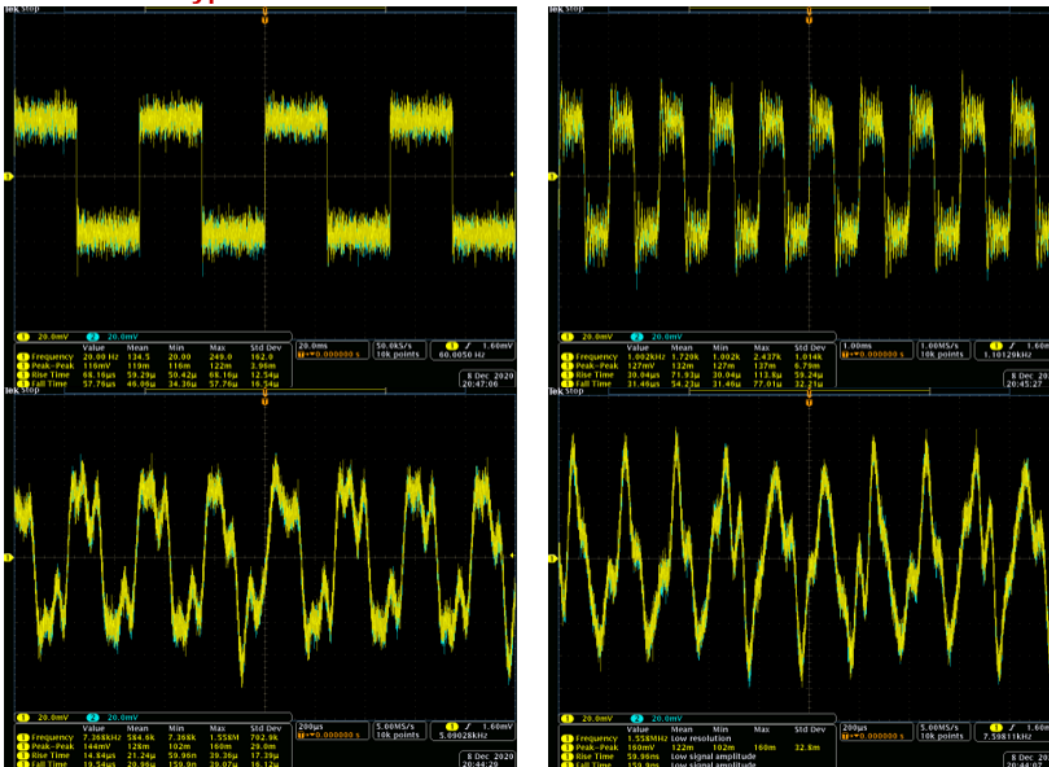


Figure 18: Triangle Waves from the Samsung S8 (Top Left) 20 Hz (Top Right) 2500 Hz (Bottom Left) 3750 Hz (Bottom Right) 5000 Hz

A simple sine wave is able to be generated and produced from the auxiliary port without issue. Whilst testing, it was discovered that depending on the hardware, the auxiliary port is able to create the different qualities of the signals depending on the range of signals. For example, the Samsung S8 was able to produce square waves quite accurately between ranges of 20 – 2500 Hz (this can be seen in Figure 18). Beyond this



Group Design Project 18: Electronics Everywhere Application Handover

range, (2500 Hz) the Samsung S8 is unable to accurately produce a square signal. The comparative square waves on the OnePlus One are shown in Figure 19.

Triangle waves work throughout the whole range for Samsung S8, the differences for the OnePlus One is that the square and triangle waves work at ranges of 20-2500 Hz and then works well for 2500 Hz and above.

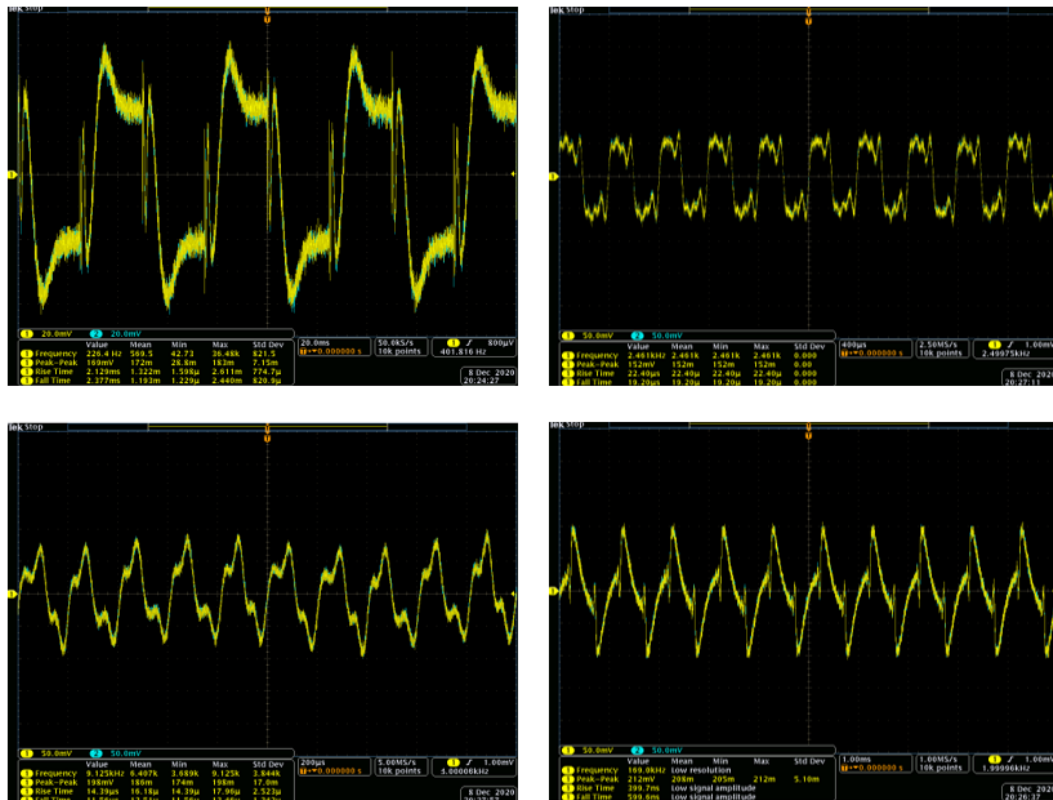


Figure 19: Square Waves from the OnePlus One (Top Left) 20 Hz (Top Right) 2500 Hz (Bottom Left) 3750 Hz (Bottom Right) 5000 Hz



## I.vi. Suggested Changes for Supporting Content



Group Design Project 18: Electronics Everywhere Application Handover

### 6 Suggested Changes for Supporting Content

#### 6.1 Instructional Changes

There is currently a document that contains a laboratory that can be used alongside the Music Mixer kit. This contains instructions for Experiment 2.2 that uses external hardware and applications. Due to the creation of the Electronics Everywhere application, we can use this application in replace (or in addition to) the hardware and software include in the original instructions.

An alternative version of the source notes has been modified to include instructions that use this application, and a small snippet of this can be seen in Figure 11. The changes made are focused around Experiment 2.2, with also an addition to Experiment 2.3. This instructional change file is included in the project archive.

#### 6.2 Website Changes

- In the future with the release of an approved application, adding information about the companion app to the kits on the website with the kit information. (schools/electronics-everywhere/)
- Updating the Music-Mixer Student Guide and Teacher Guide with information on how to use the app or a separate link with the app instructions produced with this project.
- In the Southampton University website with:
  - Information about the app when released
  - Possible update to the training videos using the app

#### 6.3 Promotional Material

To help with the process of improving the material for promotion or demonstration, the project archive includes some photographs taken of the application working with the music mixer kit. It contains both some images that are clear and for demonstration, and others that are more artistic and for aesthetics.



## I.vii.Additional Information



### Group Design Project 18: Electronics Everywhere Application Handover

## 7 Additional Information

### 7.1 A-level Physics Teachers Feedback

Along with this project, the team ran surveys that got some feedback from A-level physics teachers on the music mixer board, and the application as a whole. The feedback from these surveys are included in the project archive. A summary of key points is provided below.

#### 7.1.1 Summary of Teacher Feedback

##### 7.1.1.1 Feedback on Current Equipment and the Requirement for a Companion Application

Out of 13 A-level physics teachers with experience with the music mixer kit:

###### 7.1.1.1.1 Equipment

- 93% of teachers agree, or strongly agree, that the Music Mixer Kit is easy to plan lessons around.
- 77% of teachers agree, or strongly agree, that the Music Mixer Kit is easy for students to use and understand.
- 93% of teachers agree, or strongly agree, that students have their interest in electronics sparked during experiments and lessons involving the kits.

###### Additional comments:

- Teachers tend to use the board for the other experiments than the Music Mixer section (such as the Plank's constant) due to limitations in the availability of signal generators or oscilloscopes in classrooms
- Ability to add ammeters and voltmeters at any point throughout the board might be a useful addition
- A 5<sup>th</sup> LED for the Plank's constant section (or the ability to add an additional LED themselves)
- Showing a discharge curve of the capacitor is easy, but showing a charging curve is not

###### 7.1.1.1.2 Companion Application

- 85% of teachers agree, or strongly agree, that the class would benefit from having instructions for experiments in the app.
- 85% of teachers agree, or strongly agree, that the class would benefit from an application that is tailored for running experiments for the Music Mixer Kit.
- 93% of teachers agree, or strongly agree, that having the signals from the signal generator on the same screen (or as an overlay) as the oscilloscope will be useful to visualise the mixing of the signals

#### 7.1.2 Additional Feedback

Nearing the point of technical work, the app was handed to friends of the team with a view to get some informal feedback. After receiving this informal feedback, it seemed relevant to include, as the team members also agreed with the thoughts. At the time of



### Group Design Project 18: Electronics Everywhere Application Handover

writing, these changes could not be implemented without considerable change, and as a result were decided to be left out, since part of the specification was a polished application over one that is experimental and buggy. The following are the key points from the informal feedback:

- The text size generally throughout the app could be larger
- The navigation of the app could be clearer by having multiple “tiled” boxes that launch the user into the different parts of the kit, e.g. a tile for “exercises” and “practical” as opposed to having a bottom bar
- The descriptions of some of the tiles in the app are too long e.g. the text on the tile that launches the user into the music mixer page could be just “Music Mixer”
- It is not always apparent which tiles in the app are clickable
- The size of the tiles does not align with the importance of its content
- It wasn’t apparent that clicking outside of a pop-up would exit that pop-up
- Titles could be used on the home screen to more clearly show the different sections and their content
- The ability to scroll in some areas is not always apparent
- The buttons for “completed” and “unfinished” in the instructions didn’t add anything useful
- The theming of the instructions of the Logic and Arithmetic kit was not consistent with the rest of the application

## 7.2 Laboratory Testing

A thorough suite of testing was done in laboratories at the end of the project to try and get a deterministic behaviour of the application on different devices and what some of the problems might be. These results are in the project archive, along with images of the oscilloscope views taken of these results. This allows the reader an insight into the testing and behaviour of the app without testing in a laboratory themselves.

## 7.3 Promotional Materials

Within the project archive is an assortment of images taken of the music mixer kit working with the application in many different configurations. The idea of these images is to allow them to be used for promotional materials. Some images are there simply to be aesthetic, and others are situations that the app might be used in, that are still simple and aesthetic enough to be used in promotional or eye-catching documentation, website, video or other public domain aspects. These are not intended to show how the application works.

## 7.4 Demonstration Video and Tutorial Video

Also included in the document archive are videos and documentations that show the operation of the application, and how a user is instructed to use the application in its current state.



## I.viii. Project Structure



Group Design Project 18: Electronics Everywhere Application Handover

### 8 Project Structure

#### 8.1 Model View Controller Structure

Seen in the “lib” folder (the flutter part of the code), the project is split up into a Model View Controller (MVC) structure. This is natural with the way that flutter handles a lot of its code already. The difference between a stateless, and stateful widget require either variables to be stored (or not) within the class, with the handler functions defined outside of the view. This was something we moved a little further for the practical page, as we needed a method to correctly scope the page and ensure that the most complicated bit of code did not balloon, which would have caused maintenance issues. As a result, all the code that mentions variables that are used in the practical page, is stored in the model.dart. The functional code that changes variables in this page (or handles update events from the stream or animation events) is placed in the controller. Any of the widget display or the way in which items are rendered or displayed are within the view section of this code the graph is within this section, since the majority of the processing is handled by using the model variables, and is therefore not something that is updating variables, but instead one that is dynamically updating to display them.

#### 8.2 Project Tree

The project tree at the end of this document contains all the most important code files that are used during the project. If the file is not in the following tree, the team has not worked on it, and it was automatically generated by Flutter.



## Group Design Project 18: Electronics Everywhere Application Handover

```

Electronics Everywhere
| pubspec.lock
| pubspec.yaml
| README.md
+---android
| \---app/src/main
|         | AndroidManifest.xml
|         \---java/com/example/gdp_18
|                 MainActivity.java
|                 MicStreamPlugin.java
+---assets [Folder filled with image assets]
+---ios/Runner
|     AppDelegate.h
|     AppDelegate.m
\---lib
|     data.dart
|     enumerations.dart
|     main.dart
+---controller
|     controller.dart
+---model
|     microphone input stream.dart
|     model.dart
\---view
|     theme.dart
+---screens
|     |     home.dart
|     |     logic arithmetic.dart
|     |     music mixer.dart
|     |     settings.dart
|     +---interactive images
|     |     additionalInformationPages.dart
|     |     data.dart
|     |     interactiveboard.dart
|     |     tooltip.dart
|     \---labs
|         capacitor discharge.dart
|         plancks constant.dart
|         potential dividers ac.dart
|         potential dividers dc.dart
\---widgets
|     lab templates.dart
|     logic labs.dart
|     mixer labs.dart
|     mixer practical.dart
|     navigation bars.dart
|     ukesf branding.dart
\---music mixer practical
+---graphs
|     graphing configuration handler.dart
|     graph painter container.dart
|     wave painter.dart
+---miscellaneous
|     edit channel.dart
|     settings popups.dart
+---side bar
|     +---combined
|     |     layout selector.dart
|     |     settings bar combined.dart
|     +---oscilloscope
|     |     settings bar scope.dart
|     \---signal generator
|         custom icons.dart
|         graph icons.dart
|         layout selector.dart
|         output toggler.dart
|         parameter tabs.dart
|         settings bar sig gen.dart
|         wave parameter editor.dart
\---side bar header
|     side bar selection.dart
|     side_bar_settings.dart

```





