|  |
| --- |
| MSD Group |
| Introducing the 8x8 wireless array |
| A Quick Start Guide |

|  |
| --- |
| Kazemi, Ben  5-16-2023 |

Table of Contents

[1. Specification 2](#_Toc136350143)

[1.1. Overview 2](#_Toc136350144)

[1.2. Repository Folder 3](#_Toc136350145)

[2. Electronics 4](#_Toc136350146)

[2.1. Array board 4](#_Toc136350147)

[2.2. Controller 5](#_Toc136350148)

[2.3. Block Diagram 6](#_Toc136350149)

[2.4. Power USB C Power Delivery and Quick Charge Capabilities 7](#_Toc136350150)

[2.5. Orientation 8](#_Toc136350151)

[2.6. Communication 9](#_Toc136350152)

[2.7. Characteristics 10](#_Toc136350153)

[2.7.1. Power Characteristics 10](#_Toc136350154)

[2.7.2. Communication Characteristics 10](#_Toc136350155)

[2.8. Jumper Configuration 11](#_Toc136350156)

[2.9. SYNC 13](#_Toc136350157)

[2.9.1. Two Boards 13](#_Toc136350158)

[2.9.2. Three Boards 13](#_Toc136350159)

[2.9.3. Alternative: Three Boards 14](#_Toc136350160)

[2.9.4. Daisy Chaining More Boards 14](#_Toc136350161)

[2.10. Firmware 15](#_Toc136350162)

[2.10.1. Installing the Adafruit BSP 15](#_Toc136350163)

[3. Jig Holder 16](#_Toc136350164)

# Specification

This document acts as a quick start guide for a new wireless 8x8 board from the MSD Group at UCL.

The board and firmware are designed by Ben Kazemi.

All files will be provided in the same repository as this document.

## Overview

The output is very similar to the existing acoustophoretic boards in that instead of a 16x16 array of transducers, this board contains an 8x8 array totalling 64 transducers in a square formation.

The difference here is that the controller board that couples to the array board shadows its size, so it is scalable. Furthermore, the board contains an ESP32 which contains Wi-Fi, classic Bluetooth, and BLE wireless communication along with an ARM cortex MCU connected via UART, SPI, and 6 IO pins to an altera Cyclone IV FPGA. The last major difference is that the board is powered through QC3 or PD3 USB C protocols, that means you can get most power banks and wall plugs that connect via USB C to power this board with at a maximum of 20V.

## Repository Folder

This document is in the root folder of a repository with all necessary files.

There are 5 folders.

A screenshot of a computer

Description automatically generated with medium confidence

2D Drawings contains the block diagram and dimensions.

3D Models contains files for the Jig part, step files for both of the boards, and an inventor files for the two parts mated.

Firmware contains the MCU and FPGA firmware.

PCB Files contains the Altium files for both boards, gerber files for the V1 board, and miscellaneous files.

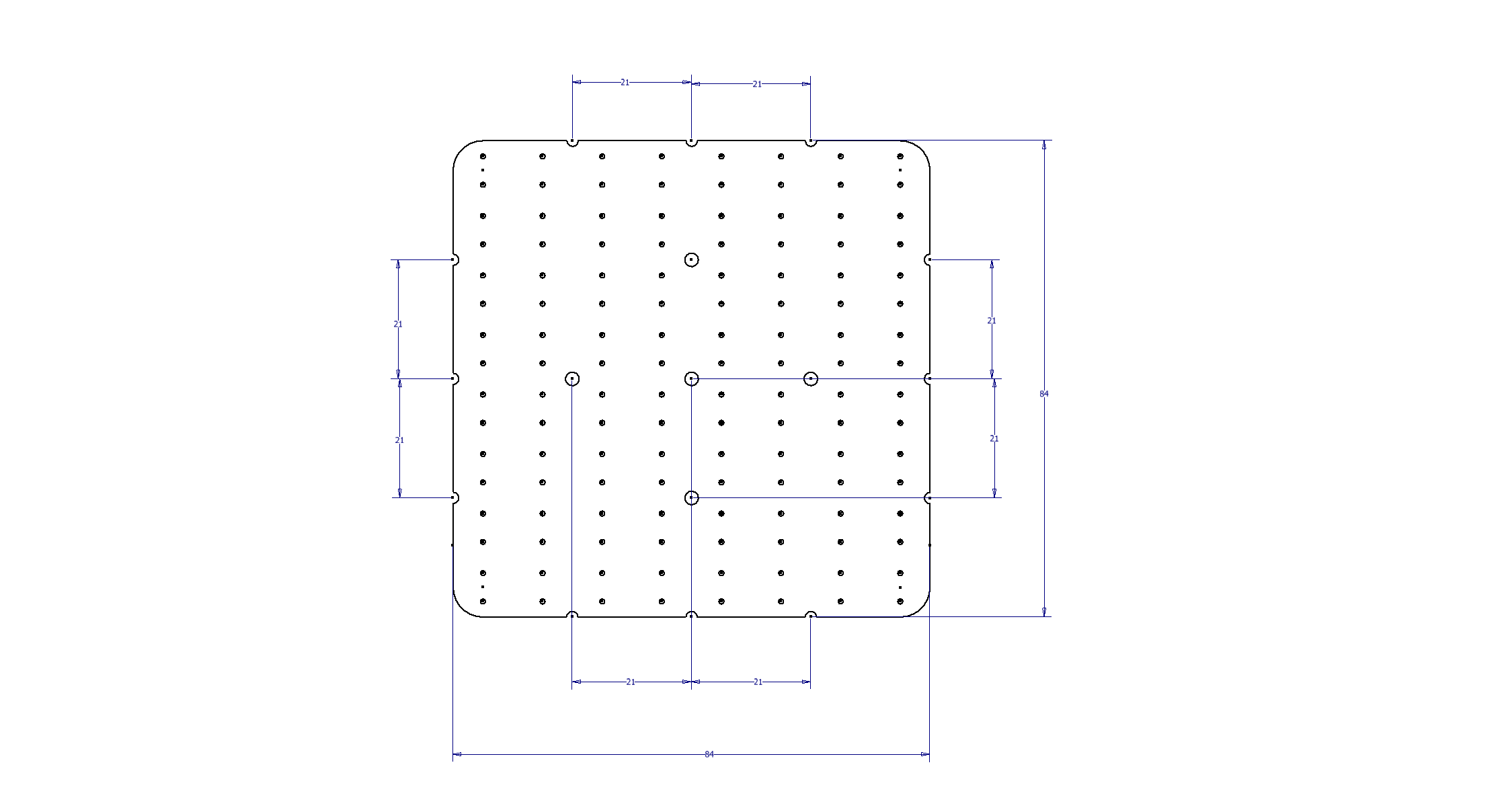
Software contains a Bluetooth Classic server written in c#, it sends and receives data from this board.

# Electronics

## Array board

Here is a dimension drawing for the edge and centre mounting holes, and board dimension.

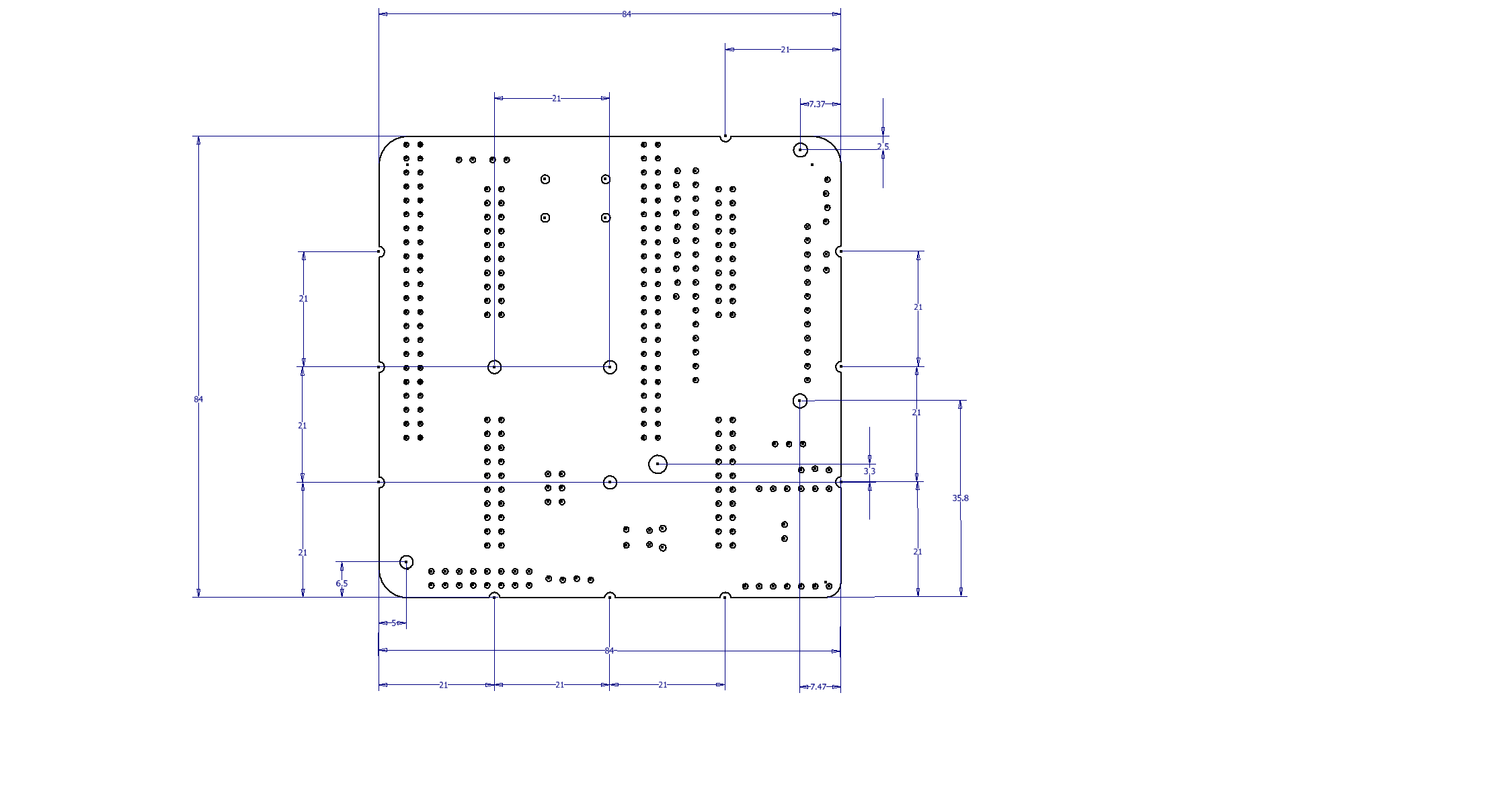
Units: mm.



## Controller

Here is a dimension drawing for the edge and centre mounting holes, and board dimension.

Units: mm.



## Block Diagram

**A screenshot of a computer

Description automatically generated with medium confidence**

We Utilise an EP4CE6 Altera Cyclone IV FPGA for parallel control of the transducer array, and a ESP32 to receive and parse wireless data, forwarding a datastream to the FGPA. The latter brings a relaxed development environment and the two are connected over 6 IO pins, along with SPI and UART.

Originally, the newer Nordic nRF52480 was chosen as it utilises the newer Bluetooth Low Energy BLE, however a move was made to the ESP32 due to its Bluetooth Classic capability, giving a much easier Bluetooth Serial Port protocol to interface with. Furthermore, after careful consideration, BLE serial transmit packets are roughly 112 Bytes maximum without further low level development and investigation, whereas Bluetooth Classic provides roughly 320 Byte packet sizes. This is important since the packet size we use is 128 bytes. This will be discussed further in the characterisation chapter.

## Power USB C Power Delivery and Quick Charge Capabilities

This board can be powered from any low noise DC power source through the Power Monitor port, this may be desirable for debugging or characterisation, however the board is designed to source power through the USB C port. The USB C port can be utilise either the power delivery PD3 protocol or the Quick Charge QC3 protocol. It is important that when selecting a power bank or power plug, that the rating is at least the desirable voltage: 12 V to 20 V, and at least 1 Amp.

Current draw at 12 V is measured at 360 mA, and 20 V at 460 mA. This is when each transducer is outputting a 40 KHz sine wave in unison.

If you intend to run this off a power bank and wish to know a rough life time, then divide the battery capacity by the above current draw.

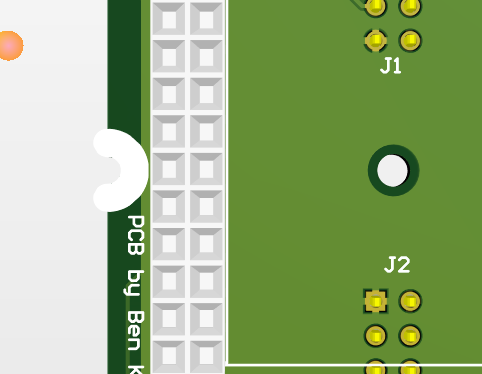
A purple modular board is used for power sources that utilise QC 3 or PD 3. You must select the desired voltage with the DIP switches, the chart is given below. Make sure the voltage you select is deliverable by your power source, it is always wise to check your voltage with a multi-meter the first few times you set up a configuration to make sure the power levels are what you’d expect.

|  |  |  |  |
| --- | --- | --- | --- |
|  | S1 | S2 | S3 |
| 5V | 1 | 1 | 0 |
| 9V | 1 | 1 | 1 |
| 12V | 1 | 0 | 1 |
| 15V | 0 | 0 | 1 |
| 20V | 0 | 1 | 1 |

## Orientation

Since there are two coupling boards, orientation is tantamount since incorrect orientation will cause electrical damage to one or more of the components.

There are white semicircle indicators on the edge of both boards, you want both marks on each board to face the same way. Newer revisions also have a large arrow to this indicator.

 A picture containing screenshot, map, diagram, circuit

Description automatically generatedA picture containing screenshot, circuit, electronic engineering, electronics

Description automatically generatedA screenshot of a computer

Description automatically generated with low confidence

## Communication

Bluetooth classic has a baud rate that is automatically set by the Bluetooth stack and is a maximum of 3 Mbps, however many factors that lower this and thus should be treated as a theoretical maximum.

The baud size used to communicate between the computer and the MCU over USB is 2 Mbps, this should be treated as a potential bottle neck when looking at the entire communication stack. It is the same as the baud rate used for the serial UART between the MCU and the FPGA. This gives a theoretical maximum throughput of 250 KB/s, note that this is separate to the throughput over Bluetooth. Bluetooth Classic provides roughly 320 Byte packet sizes. The packet size we use is 128 bytes. Therefore, there is a theoretical maximum throughput of 250 KB/s / 128 bytes = 1950 Hz, however there are many other limiting factors which are but not limited to, maximum packets sent per second, RF strength, distance, host side generation throughput, host side Bluetooth radio, MCU speed, etc. We measure a throughput of 50 Hz.

There are jumpers that you will need to configure to write firmware and to configure the MCU, but once configured, there is a micro-USB port to direct communication to the MCU, this is the Serial object in the firmware. Serial1 is the object that echo’s the read data from the Bluetooth port and sends it over UART to the FPGA.

## Characteristics

### Power Characteristics

The following measurements were made with all transducers running at 40 KHz:

12 V à 360 mA = 4.5 W

20 V à 460 mA = 9 W

And with all transducers off:

12 V à 0.06 A = 1.2 W

This means there is roughly a 6 mA draw per transducer, equal to around 0.1 W.

### Communication Characteristics

In our first unoptimized prototype, we measure a 50 Hz throughput. The baud rate is 2million on both Serial and Serial1 objects.

## Jumper Configuration

There are multiple jumpers requiring configuration on the first revision of the controller board, we will discuss them here. Note that a new revision of the controller board has a subset of these jumpers.

A screenshot of a computer

Description automatically generated with medium confidence

Above, we can see two jumpers, 3V3\_EN, and VBUS\_Jumper. 3V3\_EN needs to be carefully cut in the middle for the MCU to be enabled.

VBUS\_Jumper should be connected when you want to power the MCU through the USB C port. It should only be disconnected when you want to connect and power the MCU via micro USB to a computer, for programming or serial debugging. **Connecting this jumper, USB C port, and the micro USB port at the same time will result in parallel 5V sources and as such is a forbidden configuration.**

A picture containing text, screenshot, font, green

Description automatically generated

Here we can see the 5V\_SEL jumper. A jumper should be placed exclusively between 5V and EXT\_5V as this is the only working configuration. 5V feeds the FPGA and ESP32.

A picture containing text, screenshot, diagram, design

Description automatically generated

VS\_Jumper should be connected all the time if you want the USB C power pins to connect to VS, VS stands Voltage Source and is the voltage that feeds the transducers. You should only disconnect it if you plan on powering the board through an alternative source via the MONITOR pins.

A picture containing screenshot, diagram, text, green

Description automatically generated

Here are the Master and Slave jumpers, note in the older boards, these are in the top corner of the board. You should only connect ONE jumper to this set of jumpers, and that only ONE master can be set per set of boards, if you have a collection of boards daisy chained via the sync stage.

## SYNC

A picture containing map, plan, screenshot, circuit

Description automatically generated

There is a method to synchronise multiple boards by daisy chaining boards together through a mini-jack cable, note that **you can only have one master in a set of boards, and you cannot connect two output signals together, otherwise the FPGA will get damaged.**

Both Links are connected to two different FPGA pins. The purpose is to have multiple configurations possible, however we don’t have to use both ports as it entirely depends on the configuration and number of boards used. Here are a few example configurations.

### Two Boards

A screenshot of a video game

Description automatically generated with medium confidence

Here the connection between the master and slave utilises one cable. Note and remember how ‘master’ means you set the jumper pin named ‘master’, and likewise for ‘slave’.

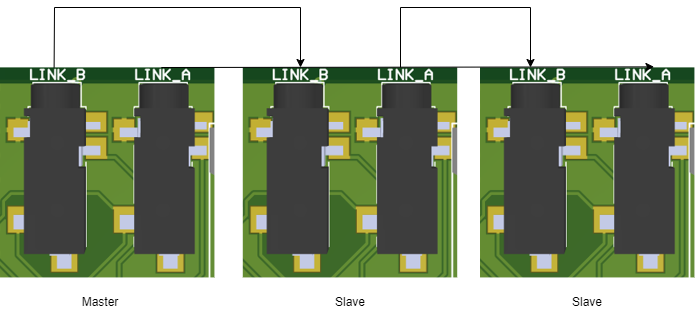
### Three Boards

A screenshot of a computer chip

Description automatically generated with low confidence

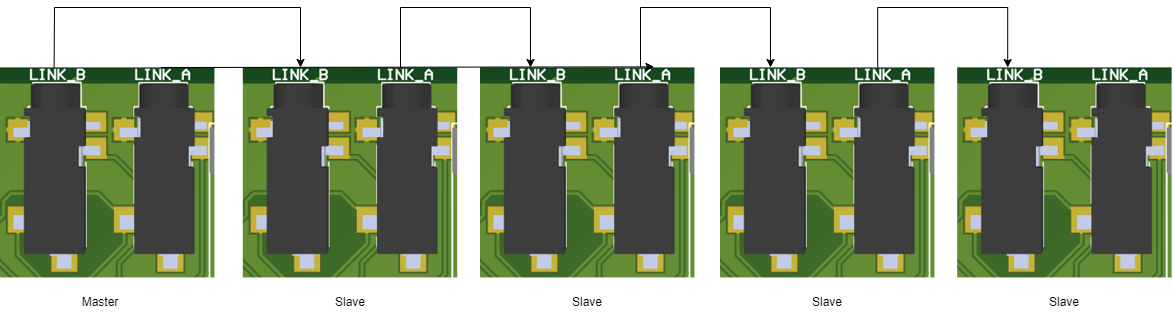
Similar to before, but now we add a third board from the second link on the master board to a second slave board, note and remember how both slave boards have the jumpers set to slave.

### Alternative: Three Boards



Notice how we can daisy chain the boards to have an alternative arrangement of three boards; this configuration can be expanded. Remember each configuration needs careful consideration with the firmware programming and jumper settings. This daisy chain configuration is not supported by the default FPGA firmware.

### Daisy Chaining More Boards



We can daisy chain multiple boards with this configuration.

## Firmware

There are two sets of firmware to be used, one on the FPGA and one on the MCU. The MCU simply receives wireless data, and forwards it to the Serial port connected to the FPGA.

The FPGA firmware receives this data and decodes the packets appropriately to set the phase and amplitude accordingly for each transducer.

Each firmware can be found in the repo folders. Note you may only need to use the quartus programmer software to program the necessary jic file.

### Installing the Adafruit BSP

The ESP32 requires its own board files. Follow the instructions here for a guide on the ESP32 board that is used.

<https://learn.adafruit.com/adafruit-huzzah32-esp32-feather>

To quickly get it up and running in an Arduino environment, add this url to the board url’s in your preferences:

<https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_dev_index.json>

## Bill of Materials

### Array BOM

|  |  |  |  |
| --- | --- | --- | --- |
| **Comment** | **Designator** | **Footprint** | **Quantity** |
| EMIT | 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64 | TRANSDUCER | 64 |
| 0.1 uF | C1.1, C1.10, C1.11, C1.12, C1.13, C1.14, C1.15, C1.16, C1.17, C1.18, C1.19, C1.2, C1.20, C1.21, C1.22, C1.23, C1.24, C1.25, C1.26, C1.27, C1.28, C1.29, C1.3, C1.30, C1.31, C1.32, C1.4, C1.5, C1.6, C1.7, C1.8, C1.9 | CAP0805 - CERAMIC | 32 |
| B27c-2x10-BSAB1-G | J1, J2, J3, J4 | B27C SMT2X10 | 4 |
| MOSFET Driver | U1.1, U1.10, U1.11, U1.12, U1.13, U1.14, U1.15, U1.16, U1.17, U1.18, U1.19, U1.2, U1.20, U1.21, U1.22, U1.23, U1.24, U1.25, U1.26, U1.27, U1.28, U1.29, U1.3, U1.30, U1.31, U1.32, U1.4, U1.5, U1.6, U1.7, U1.8, U1.9 | MIC4127YME | 32 |

### Controller BOM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Comment** | **Description** | **Designator** | **Footprint** | **Quantity** |
| OKI-78SR-5 | Voltage Regulator | 5VREG | OOKA-78SR | 1 |
| GRM32ER71A476ME15L |  | C1 | EIA6032-NOM | 1 |
| 0.33uF |  | C2 | CAP0805 | 1 |
| Waveshare FPGA |  | EP4CE6 | EP4CE6 Waveshare FPGA | 1 |
| FPGA IO |  | FPGA IO | 2X3 | 1 |
| TSSH-110-01-L-D |  | J1, J2, J3, J4 | SAMTEC\_TSSH-110-01-L-D | 4 |
| Debug access pins |  | J5 | 1X10 \_ LOCK - header | 1 |
| Pulled out pins | Debug access pins | J6 | 1X04\_LOCK - header | 1 |
| 35RASMT4BHNTRX | Sync port A, Sync port B | LINK\_A, LINK\_B | SWITCHCRAFT\_35RASMT4BHNTRX | 2 |
| Connect for Master |  | Master | 1X02\_LOCK\_ | 1 |
| Access to power rails |  | MONITOR | 2X8 | 1 |
| Adafruit Feather nRF52480 |  | ESP32 Feather | ESP32 Feather | 1 |
| Sparkfun STUSB4500 | Replace with purple board | PD SINK | Sparkfun STUSB4500/ OR USB C PD Power Delivery Trigger / Decoy Module 5V, 9V,1 2V, 15V, 20V | 1 |
| 3K BOURNS CRT0805-BY-3001ELF |  | R1 | RES-0805 | 1 |
| 10K |  | R2 | RES-0805 | 1 |
| Connect for Slave | Header, 2-Pin | Slave | 1X02\_LOCK\_ | 1 |
| Short to connect 5V to VBUS of feather | Header, 2-Pin | VBUS\_Jumper | 1X02\_LOCK\_ | 1 |
| Red WURTH 150080RS75000 |  | VS | LED-0805 - RED | 1 |

## Outsourcing

To outsource the procurement and assembly of this project, you should contact your company of choice and prepare to hand over ideally the Altium files for both boards. You only need to hand the PCBDOC file and if requested, the SCHDOC file.

You should also hand over the BOMs above.

You can provide any components you may have such as the bare board PCB or transducers.

Transducers can be purchased here <https://www.hcspeakerbuzzer.com/>

Assembly services can be found here http://www.interface2.co.uk/

# Jig Holder

A jig was designed to hold the complete board so the transducers are facing upright. This may be useful for debugging and testing. There is only one orientation possible with this jig with as shown in the pictures below.

A picture containing LEGO

Description automatically generated with medium confidence A picture containing electronic engineering, electronic component, circuit component, electronics

Description automatically generated

A picture containing playground, metal, yellow

Description automatically generated