EE 344: Electronics Design Lab

OpenBCI based EEG Acquisition System

Group: TUE-06
Aayush Rajesh (200070001)
Aditya Sriram (200070004)
Shivam Patel (200070077)

Department of Electrical Engineering
Indian Institute of Technology, Bombay

Contents

Introduction

EEG is a non-invasive method of capturing brain signals. The device consists of two parts: recording electrodes and data-capturing electronics. The electrodes are placed on the subject's head to record different spatial locations on the brain. Each electrode corresponds to a single of data. For example, the OpenBCI Cyton board provides access to 8 channels of data. The Daisy expansion board offers support for an additional eight channels. The goal is to scale the existing design to accommodate 24 channels for better spatial resolution. The design will be robust to ambient noise (including the 50 Hz power supply interference) and has the potential to be used for a wide variety of applications, including medical diagnosis and brain-computer interfacing. The deliverables of the project will include the following,

1. A custom-designed PCB: The PCB will support 24 EEG channels, provide a Wi-Fi module for communicating with a laptop/computer/phone for real-time streaming, a micro-SD card for local storage, and accelerometer support for removing noise due to head motion
2. An EEG headset with electrodes placed at spatial locations recommended by the 10-20 international standard
3. A laptop/phone application to view the data in real-time

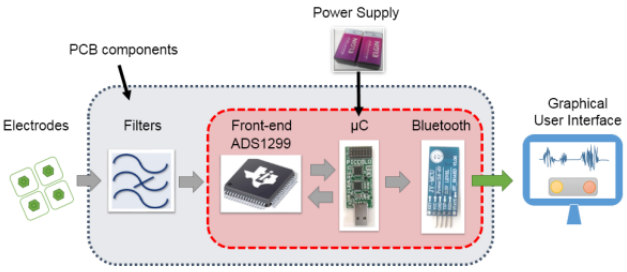


Figure 1: Existing OpenBCI design

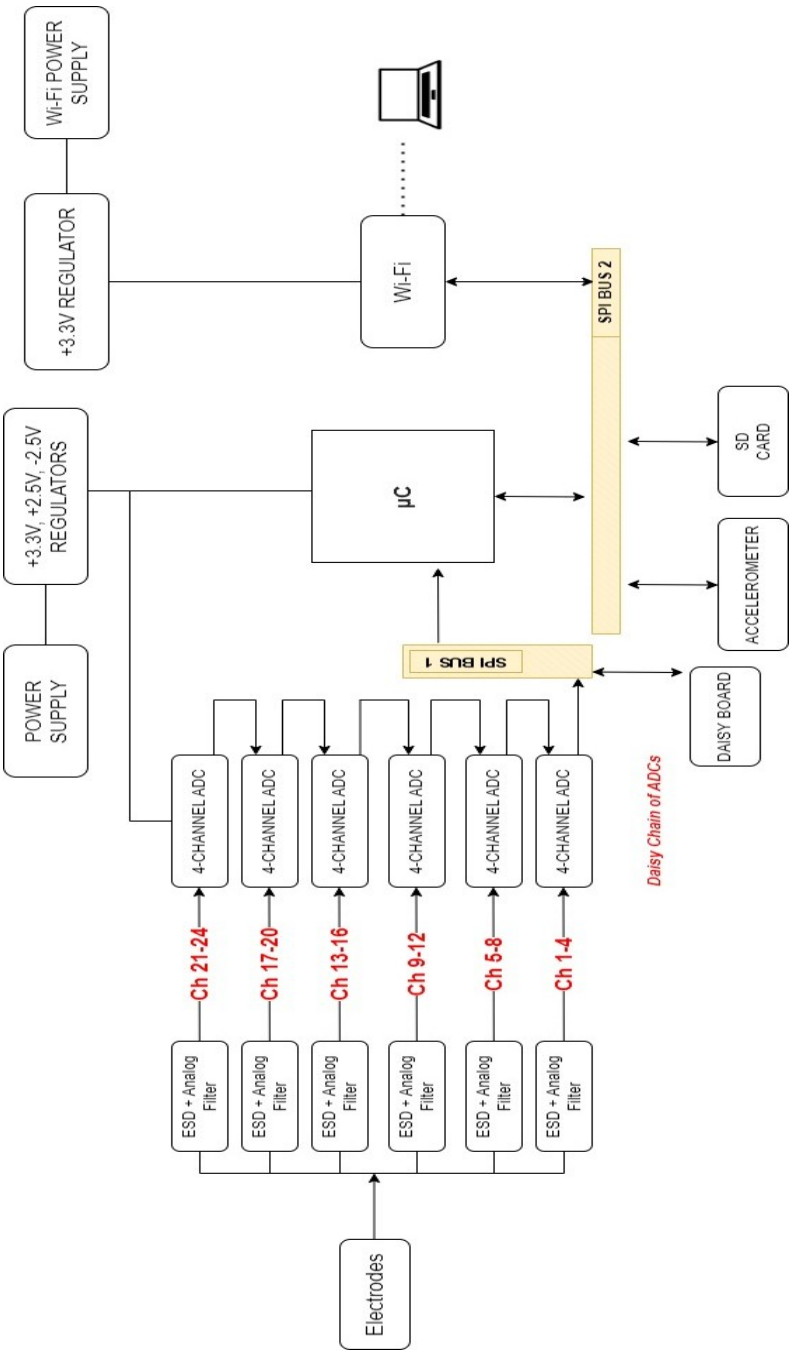


Figure 2: Design Block Diagram

Design Description

The OpenBCI Cyton Board is an 8-channel biosensing board. The existing setup takes in 8 analog inputs from the electrodes and passes them through an 8-channel A/D Converter. The digital output is sent to the microcontroller after which it is passed onto an RFduino Bluetooth Transmitter which transmits the data to a nearby device for viewing.

While scaling up the existing design to meet our requirements, there are multiple factors that we have taken into consideration. Firstly, our targeted 24-channel design would require 6 A/D converters to process the data provided by the electrodes. These devices will peripherally interface with the microcontroller, and therefore, our choice of the microcontroller will have to reflect this additional interfacing. Lastly, limitations on data rate across Bluetooth would impair the performance of our device, especially considering the larger amount of data our module targets to transmit as compared to the original design. To this end, we will look to transmit our data over Wi-Fi for better performance.

A high-level overview of our design is depicted in Figure 2. The subsections that follow give a brief overview of the role and functionality of each module and mention any changes from the existing design that we look to implement.

Electrodes

The EEG electrodes form the analog front end of the EEG processing pipeline and are of different types, including active, passive, dry, and sponge. We will be using the dry spikey and flat electrodes as these have very little preparation time. The electrodes will be mounted on a custom 3D-printed headset, adhering to the 10-20 international standard for electrode placement.

Due to an increased electrode count, specific brain regions can be targeted to study functions like motor function, sensation, and memory. We will specifically target the frontal lobe's motor and visual cortex.

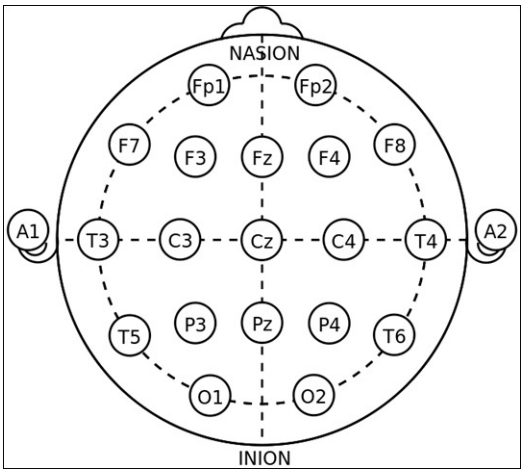


Figure 3: Electrode locations for EEG recording

A/D Converters

The A/D converters are a vital part of the analog front end. In particular, we will use the ADS1194, a low-power, 4-channel, 16-bit analog-to-digital converter for biopotential measurements. One of the major requirements for the A/D converter was the inclusion of amplifiers as the EEG signals are of the order of $10\mu V - 100\mu V$, so we have chosen the ADS1194 which has in-built programmable gain amplifiers.

After being passed through an ESD protection unit, electrode signals from 4 EEG channels are fed into the A/D converter that consists of four simultaneous samplings followed by delta-sigma ($\Delta\Sigma$) converters.

Since the design requires 24 electrodes, 6 A/D converter chips will be on the final PCB. The A/D converters will be connected in the Daisy-chain configuration to ensure synchronisation between them. The output of all the converters is multiplexed into a single stream sent to the microcontroller over a serial interconnect (SPI). The data output sent on the `DOUT` pin consists of 24 control bits followed by 24 sets of 16-bit quantized EEG data.

Microcontroller

The microcontroller at the heart of the system is responsible for gathering data from the A/D Converters and the accelerometer, processing it and passing the final values to the Wi-Fi and SD Card modules for transmission and storage, respectively. The original design for the Cyton board uses a 28-pin PIC32MX250F128B micro-controller, which comes with two I^2S/SPI modules

for codec and serial communication, up to 13 channel 10-bit A/D Converters, and up to 19 I/O pins. Since the microcontroller will now be interfacing with 6 A/D Converters, the SD Card reader, and the Wi-Fi Module, we will be using the PC32MX250F128D micro-controller, which has a larger number of I/O pins (31), which can be reconfigured as chip select lines for the additional A/D Converters.

The microcontroller will be programmed using a chipKIT UDB32-MX2-DIP bootloader, with code written in the Arduino software library.

Power Supply

The components require a Digital Supply Voltage (`DVDD`) of $+3.3V$, an Analog Supply Voltage (`AVDD`) of $+2.5V$ and an Analog Ground Voltage (`AVSS`) of $-2.5V$. AP2112K-3.3TRG1 is a fixed LDO Voltage Regulator used to generate the `DVDD` signal using the input `RAW` signal. This `DVDD` signal is fed into the LM2664 Switched Capacitor Voltage Converter, which generates the `-RAW` signal. This `-RAW` signal is further passed through the TPS72325 Negative-output Linear Regulator to generate the `AVSS` signal. The `AVDD` signal is generated by passing the `DVDD` signal through the AP2112K-2.5TRG1 Voltage Regulator.

Since the Wi-Fi has high current requirements for the Digital supply regulator, we will be using a separate Voltage Regulator (AP2112K-3.3TRG1) to generate the `DVDD` signal using the input `RAW` signal coming from another battery.

The main consideration while choosing the regulators was the required output voltage and the current rating. The Wi-Fi module requires a large amount of current while transmitting data, so we choose the $600mA$-rated AP2112K-3.3TRG1.
We will use two 3-6V DC Batteries, one as the power source for the board and another for the Wi-Fi.

Accelerometer

The LIS3DHHTR is a three-axis accelerometer with digital I^2C/SPI serial interface standard output, capable of 16-bit data output. The purpose of the accelerometer is to remove artifacts in the EEG signal due to head movements. The data returned by the accelerometer is a good baseline to reconstruct what happened in the user's recording session.

The accelerometer can also be used as a marker for different phases of experimentation. Without the accelerometer, one would have to reset the data logging software. With the accelerometer, one can simply tap the board a few times

and the created artifact would be easy to observe in the accelerometer's data stream.

SD Card

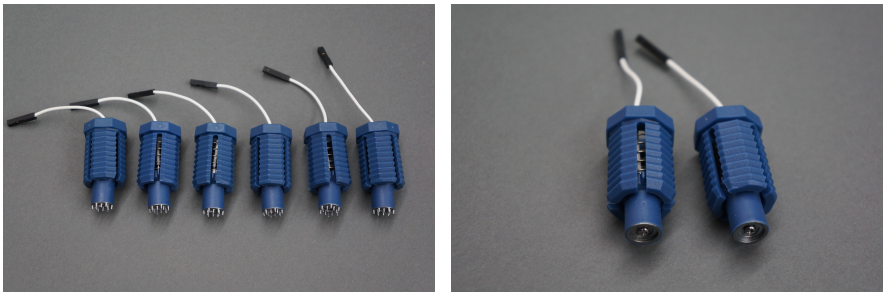
SD card is a necessary provision for logging data to local storage. This is useful in sleep study applications or when it is difficult to make wired connections to the PC. We will be using a Suntech ST-TF-003A SD card holder for the design. The data saved to the SD card is sampled at 250 Hz. This amounts to 3 MB of data per minute and hence, a high-speed SD card with large storage will be used (8 GB, 16 GB, or 32 GB). Data from the A/D converter will be sent to the SD card over an SPI bus.

Wi-Fi Module

Unlike the original design, a BLE module will not be included because of the requirement for frequent firmware updates. Another disadvantage of BLE is that the hardware must support the BLE protocol, which requires data packetization. On the other hand, Wi-Fi is based on a stream protocol which is easier to implement. OpenBCI provides a Wi-Fi shield in addition to the Cyton board, which has been known to suffer from packet losses and cyclical noise. We aim to overcome these defects by providing a reliable data transfer interface using an ESP-12S: ESP8266 Wi-Fi Module, which is Arduino compatible.

Headwear

The Ultracortex Mark IV is a device developed by the OpenBCI company, which allows users to measure and record brain activity (EEG). The following are the main components of the headset: Cables, Spikey units, Flat units, Comfort units, and Ear Clips. Unlike the original design, we have to use 8 more spikey units, i.e., 22 spikey units in total and 2 flat units. In case we also connect the Daisy module, we can use a total of 32 dry electrodes mounted on the headwear frame.



(a) Spikey Units (b) Flat Units

Figure 4: Electrodes used for EEG recording

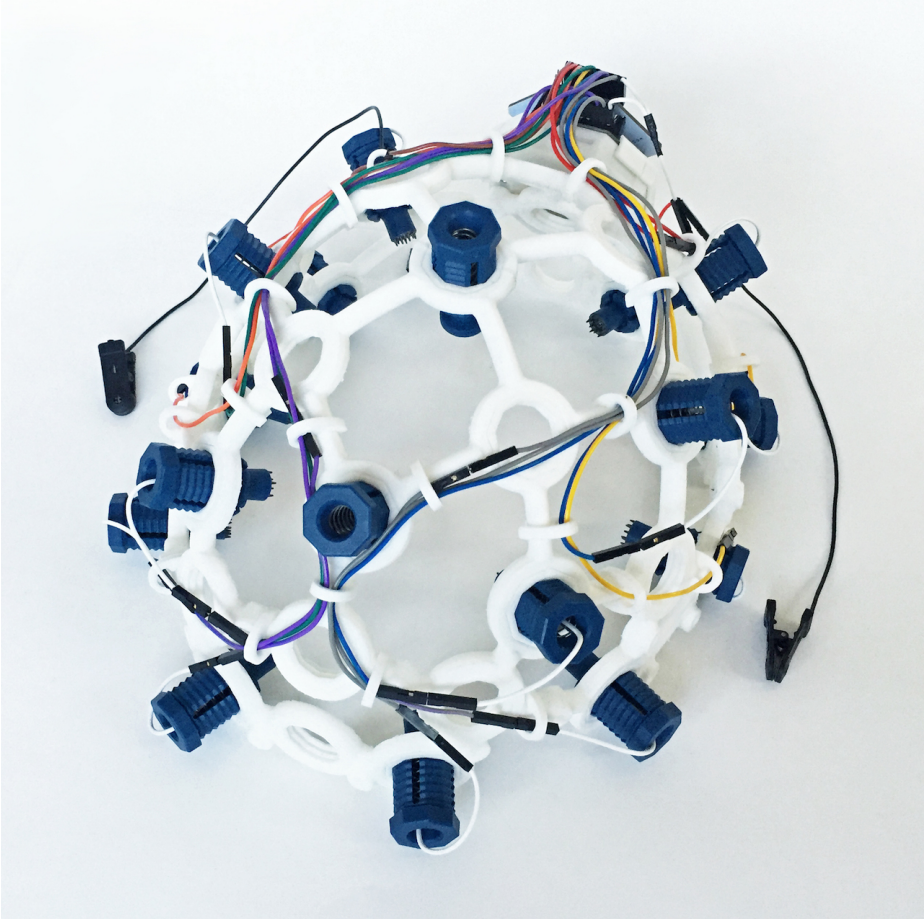


Figure 5: Assembled headwear with electrodes

Principle of Operation

ADC Topology

The original OpenBCI design had each of the ADCs communicate to the microcontroller through a single SPI bus. This was achieved by having each device send its data on the bus by taking turns. In order to implement this, each device had a designated chip select signal, and the microcontroller would selectively turn on a single chip select as needed. However, this scheme has drawbacks in the form of channel data being asynchronous. Furthermore, preliminary analysis of the firmware code suggests that it may be possible to implement a design wherein all the ADCs may be simultaneously selected to transmit their data.

We will look to assemble our ADC subsystem in a **Daisy chain** topology. The Daisy chain is an efficient means to assemble a large number of ADCs in order to obtain high channel counts. In a daisy chain, a single chip select from the microcontroller is shared by all the participating devices. The output of each ADC (D_{OUT}) is connected to the $DAISY_{IN}$ of the succeeding ADC. The result of this chain is that the final ADC in the chain outputs a stream of data corresponding to the outputs of all the preceding ADCs as well as its own data. Therefore, we only require this single ADC to communicate with the microcontroller through an SPI bus.

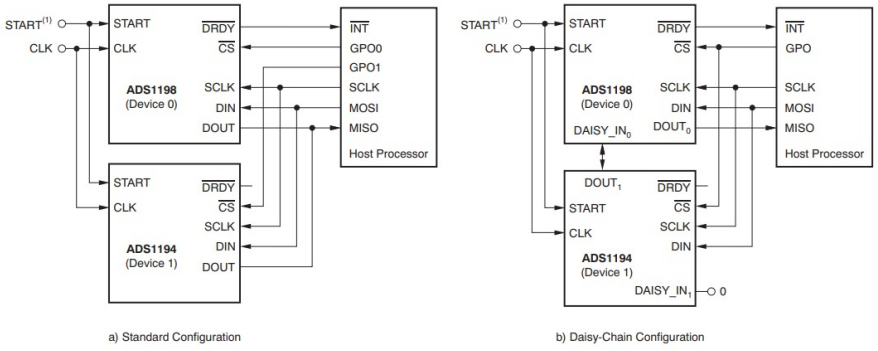


Figure 6: Comparison between standard operation and Daisy operation

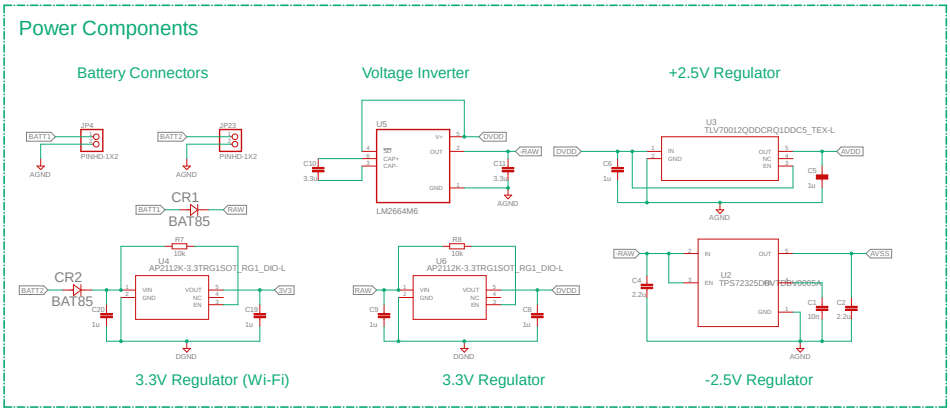
Microprocessor Interfacing

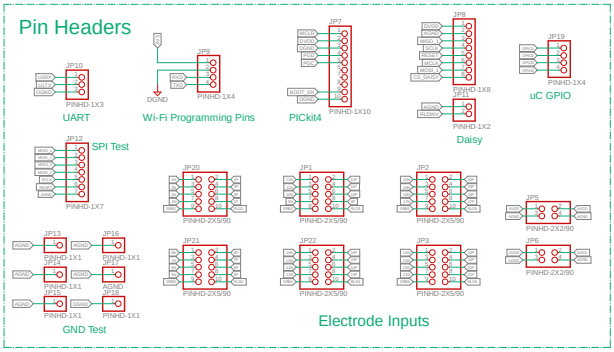
In the current iteration of the design, we have used 6 ADCs, an SD card module, an accelerometer, a WiFi module, and design inclusions for an external Daisy board interfacing. We plan on using four SPI buses for the above mentioned submodules. The breakup of SPI bus resource allocation is as follows-

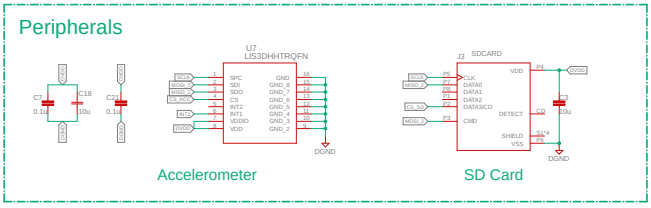
- **First SPI Bus** - For the six daisy-chained ADCs and External Daisy board interfacing
- **Second SPI Bus** - For the Wi-Fi module and peripherals such as SD Card slot and Accelerometer
- **PGEC and PGED** - For debugging and programming the controller

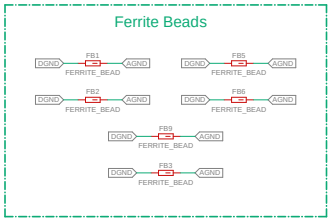
Circuit Schematic

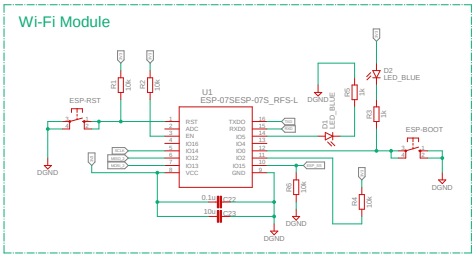

PIC32 Microcontroller
U28
PIC32MX250F128D-50I/PTTQFP44_MC_MCH-L
PIC-BOOT
R64
470k
C124
R63
1k
R67
0
D3
LED_BLUE
MCLR/RESET
R66
1k
C151
0.1u
R61
10k
RPB9/SDA1/CTED4/PMD3/RB9
RPC6/PMA1/RC6
RPC7/PMA0/RC7
RPC8/PMA5/RC8
RPC9/CTED7/PMA6/RC9
VSS_2
VCAP
PGED2/RPB10/D+/CTED11/RB10
PGEC2/RPB11/D-/RB11
VUSB3V3
AN11/RPB13/CTPLS/PMRD/RB13
PGED/TMS/PMA10/RA10
PGEC/TCK/CTED8/PMA7/RA7
CVREFOUT/AN10/C3INB/RPB14/VBUSON/SCK1/CTED5/RB14
AN9/C3INA/RPB15/SCK2/CTED6/PMCS1/RB15
AVSS
AVDD
MCLR
PGED3/VREF+/CVREF+/AN0/C3INC/RPA0/CTED1/PMD7/RA0
PGEC3/VREF-/CVREF-/AN1/RPA1/CTED2/PMD6/RA1
PGED1/AN2/C1IND/C2INB/C3IND/RPB0/PMD0/RB0
PGEC1/AN3/C1INC/C2INA/RPB1/CTED12/PMD1/RB1
RPB8/SCL1/CTED10/PMD4/RB8
RPB7/CTED3/PMD5/INT0/RB7
VBUS
RPB5/USBID/RB5
VDD
VSS
RPC5/PMA3/RC5
RPC4/PMA4/RC4
AN12/RPC3/RC3
TDI/RPA9/PMA9/RA9
SOSCO/RPA4/T1CK/CTED9/RA4
SOSCI/RPB4/RB4
TDO/RPA8/PMA8/RA8
OSC2/CLKO/RPA3/RA3
OSC1/CLKI/RPA2/RA2
VSS_3
VDD_2
AN8/RPC2/PMA2/RC2
AN7/RPC1/RC1
AN6/RPC0/RC0
AN5/C1INA/C2INC/RTCC/RPB3/SCL2/PMWR/RB3
AN4/C1INB/C2IND/RPB2/SDA2/CTED13/PMD2/RB2
R65
1M
XTAL1
C125
15p
C126
15p
C146
0.1u
C147
0.1u
C150
0.1u


Power Components
Battery Connectors
Voltage Inverter
+2.5V Regulator
3.3V Regulator (Wi-Fi)
3.3V Regulator
-2.5V Regulator

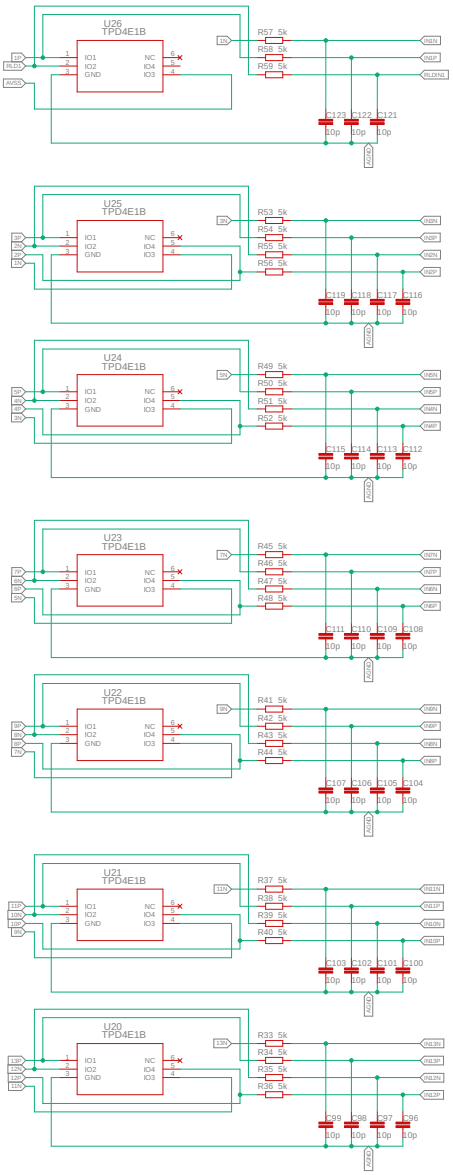

Pin Headers
UART
Wi-Fi Programming Pins
PICkit4
Daisy
uC GPIO
SPI Test
GND Test
Electrode Inputs


Peripherals
Accelerometer
SD Card


Ferrite Beads

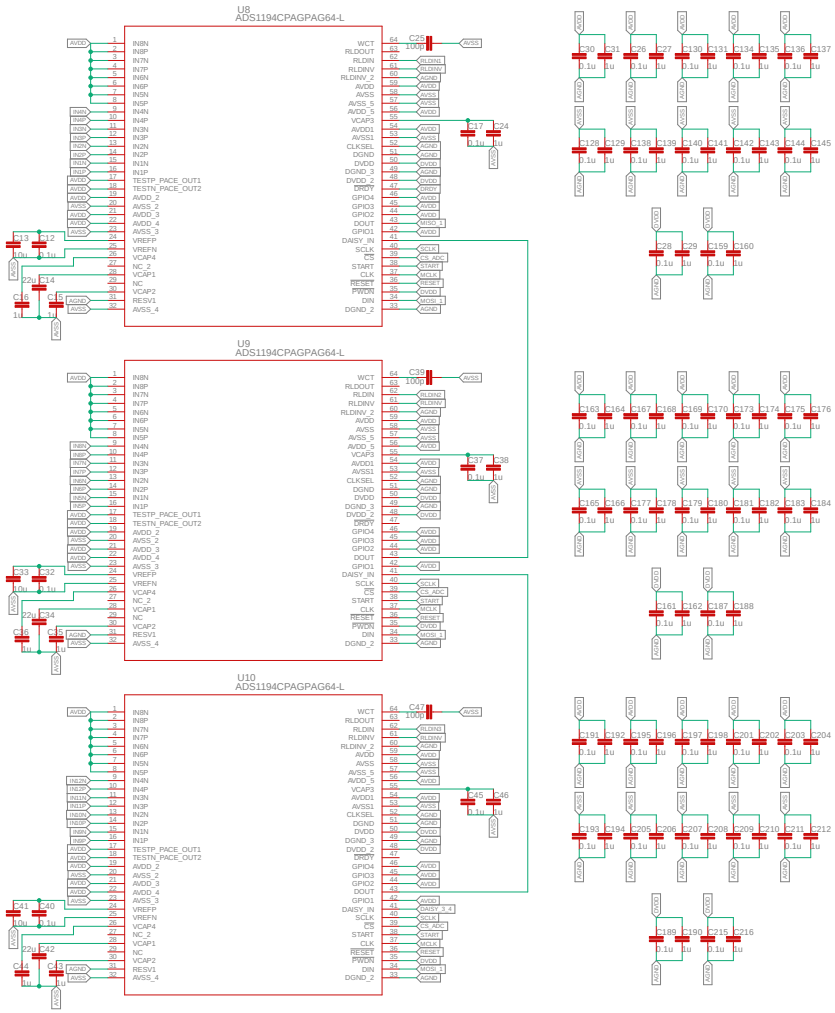

Wi-Fi Module

ESD Protection



Daisy-Chain ADCs (Part 1)

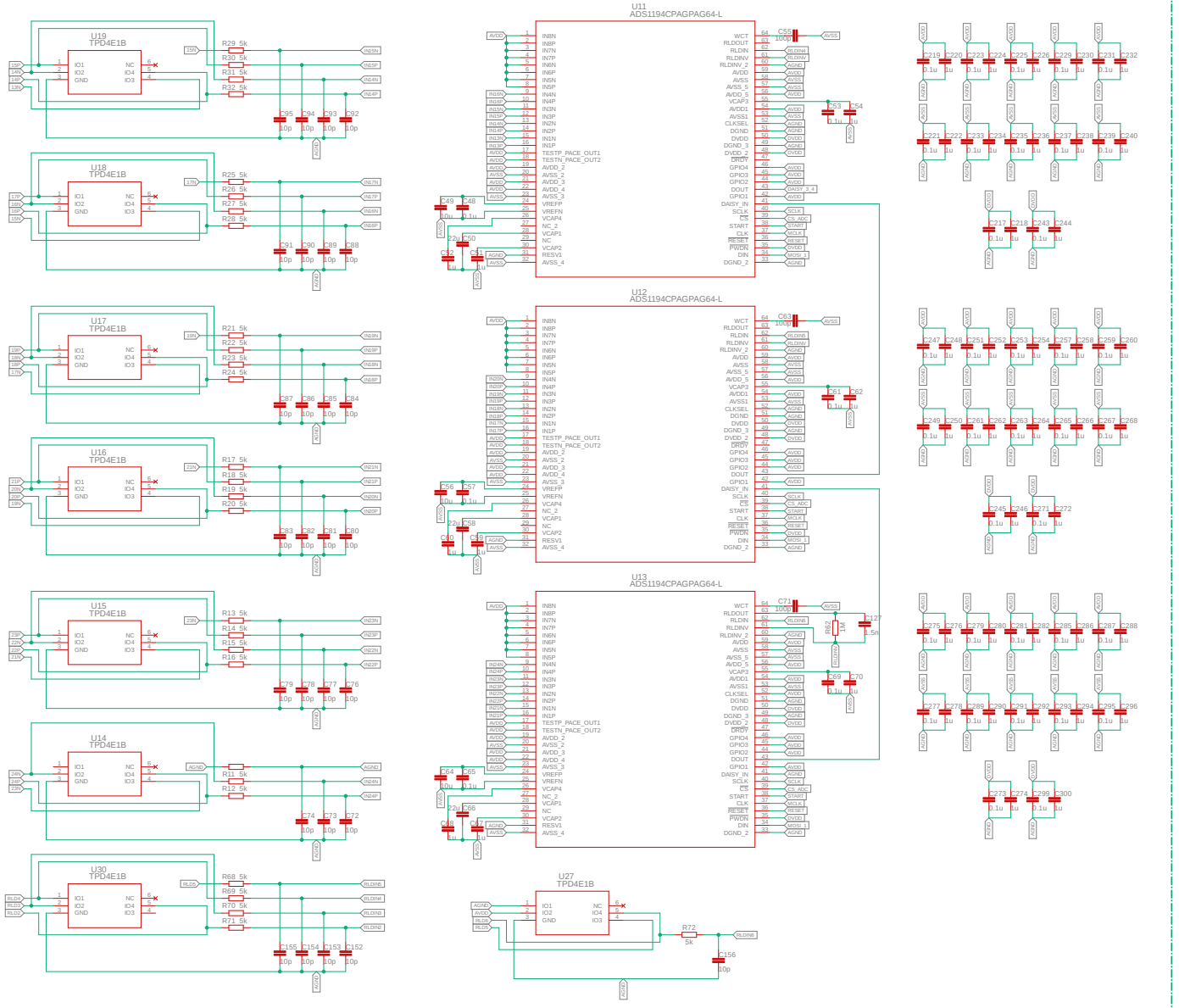
ADS 1194



ESD Protection

Daisy-Chain ADCs (Part 2)

ADS 1194



PCB Layout

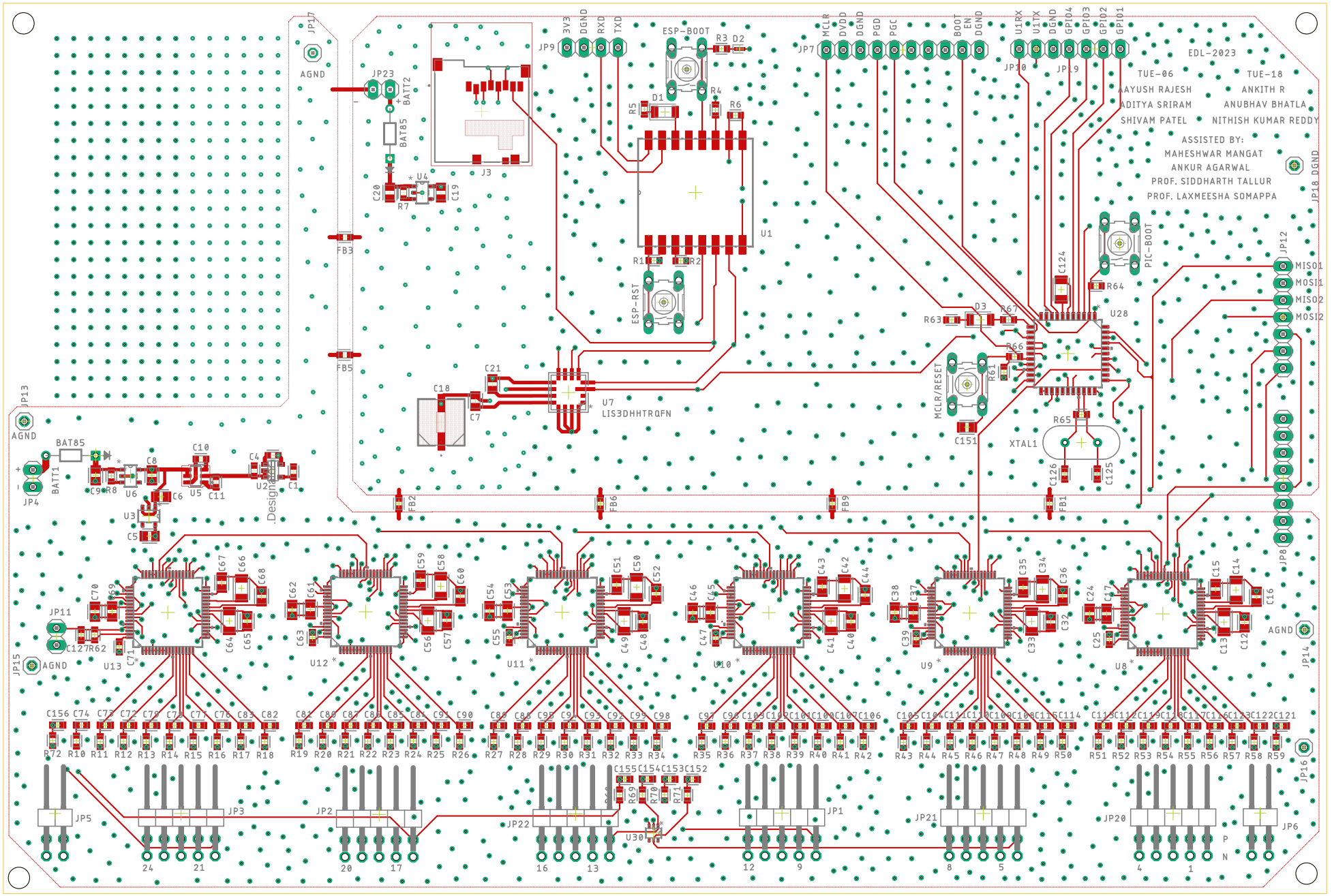
Due to the large-scale nature of our design, we have adopted a 4-layer PCB design. The top and bottom layers are where we house our components, and where the majority of the routing takes place. The inner layers are meant to function as power layers while allowing for routing as well, wherever it is not possible to do so using the outer layers. However, we have made sure to use these layers for routing as sparingly as possible, so as to ensure no deterioration in power signal quality.

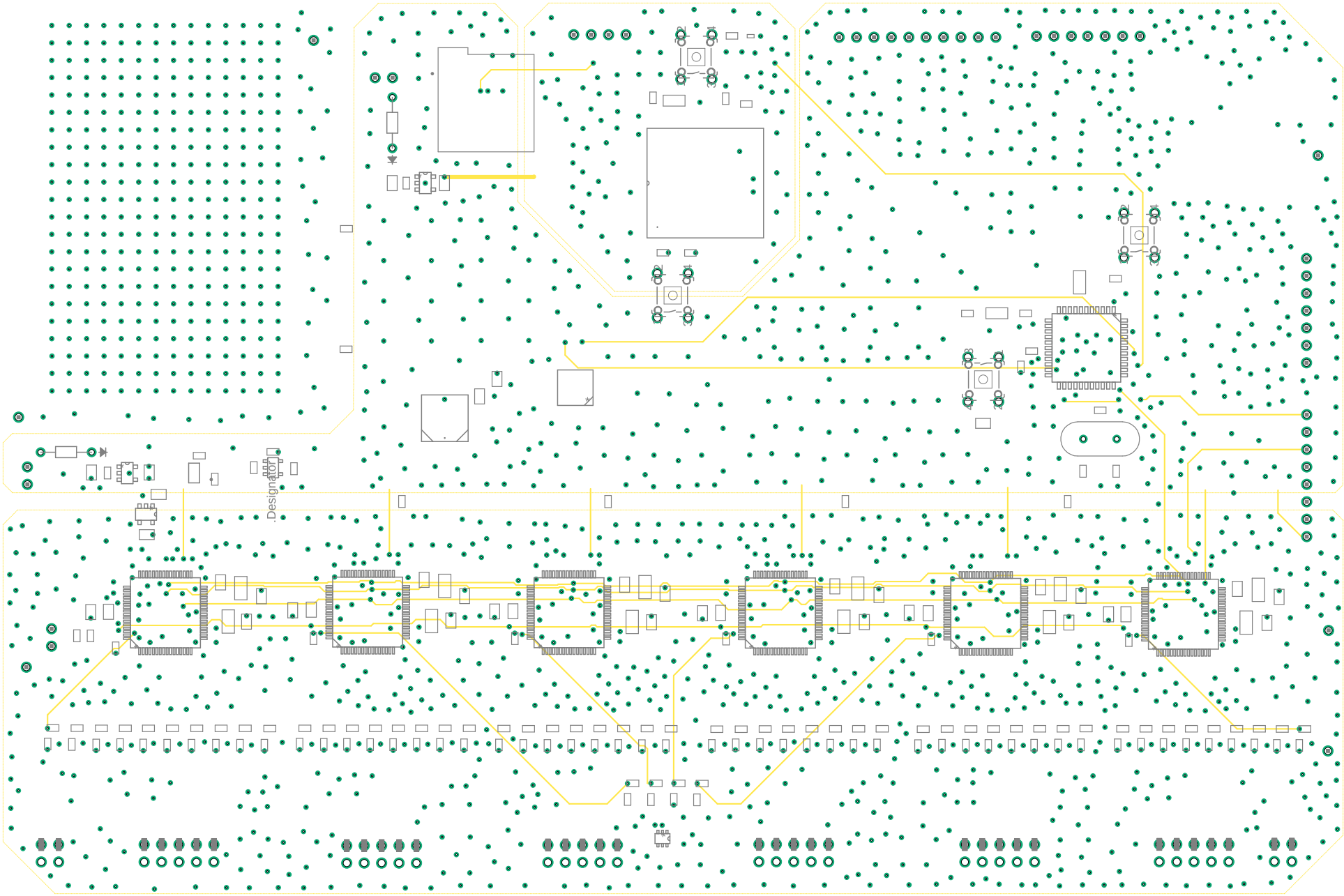
Keeping in mind our requirements, as well as the manufacturer (PCBPower) specifics, we have vias in our design of type 1-2, 1-3, and 1-4. This allows connection between the top and bottom layers, as well as connections of the top and bottom layers with the inner layer, for both routing and power signal purposes. Stitching vias, which are through-hole vias, populate the board in regions wherever we have no components or routing, in order to connect our top and bottom layer copper fills. We have also provided a prototyping region in the top-left area of our board, where we do not have any components or copper fills.

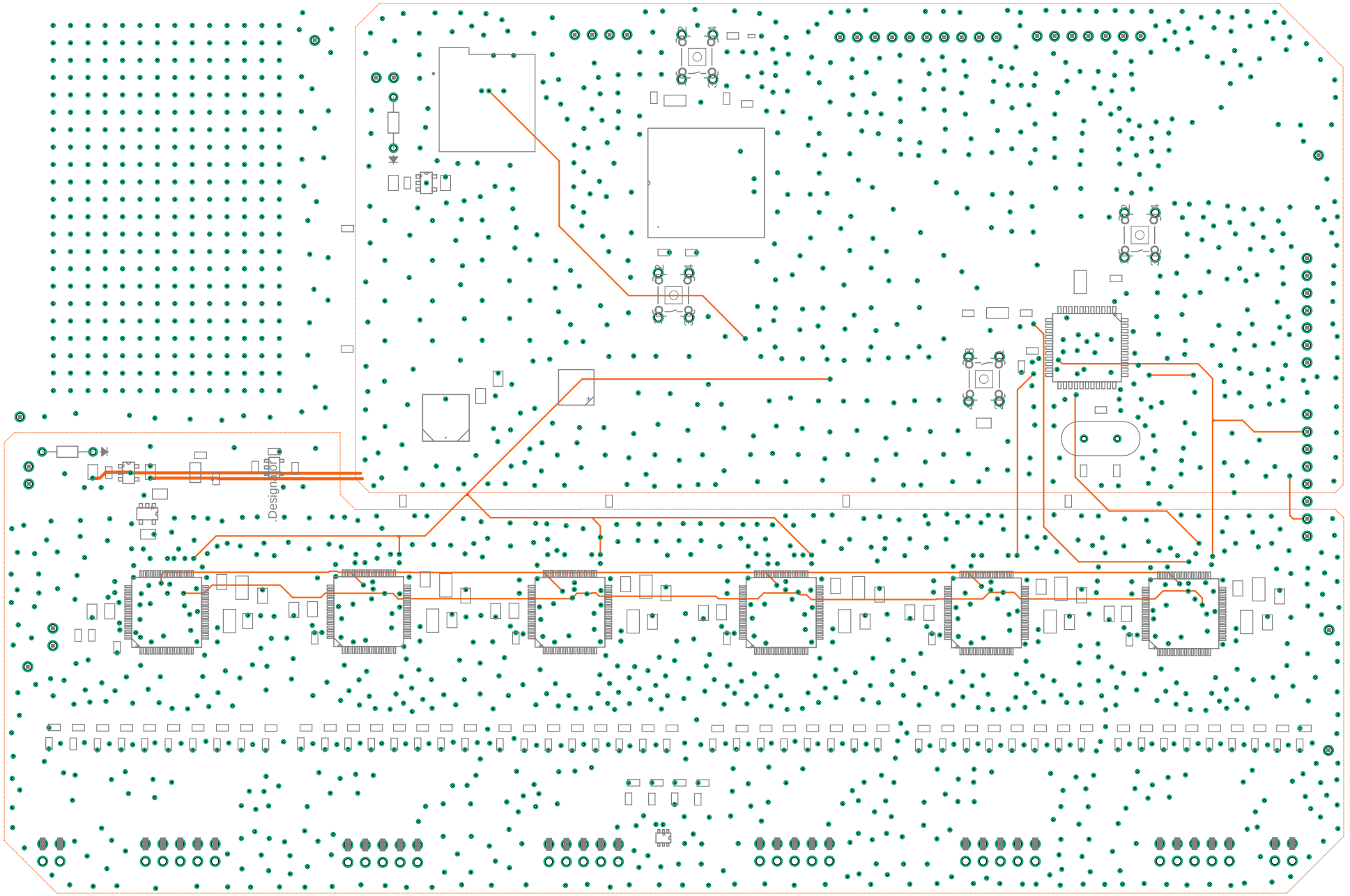
Coming to the copper fills, we have separate copper fills on the 4 layers corresponding to the Analog portion of the board (lower half) and the Digital portion of the board (upper half). Table 1 shows the exact nature of our layer stack. The demarcation of the exact regions can be seen on the following pages, which are arranged from the Top Layer all the way to the Bottom Layer.

Layer	Analog	Digital
Top	AGND	DGND
Layer 2	AVDD	DVDD
Layer 3	AVSS	DGND
Bottom	AGND	DGND

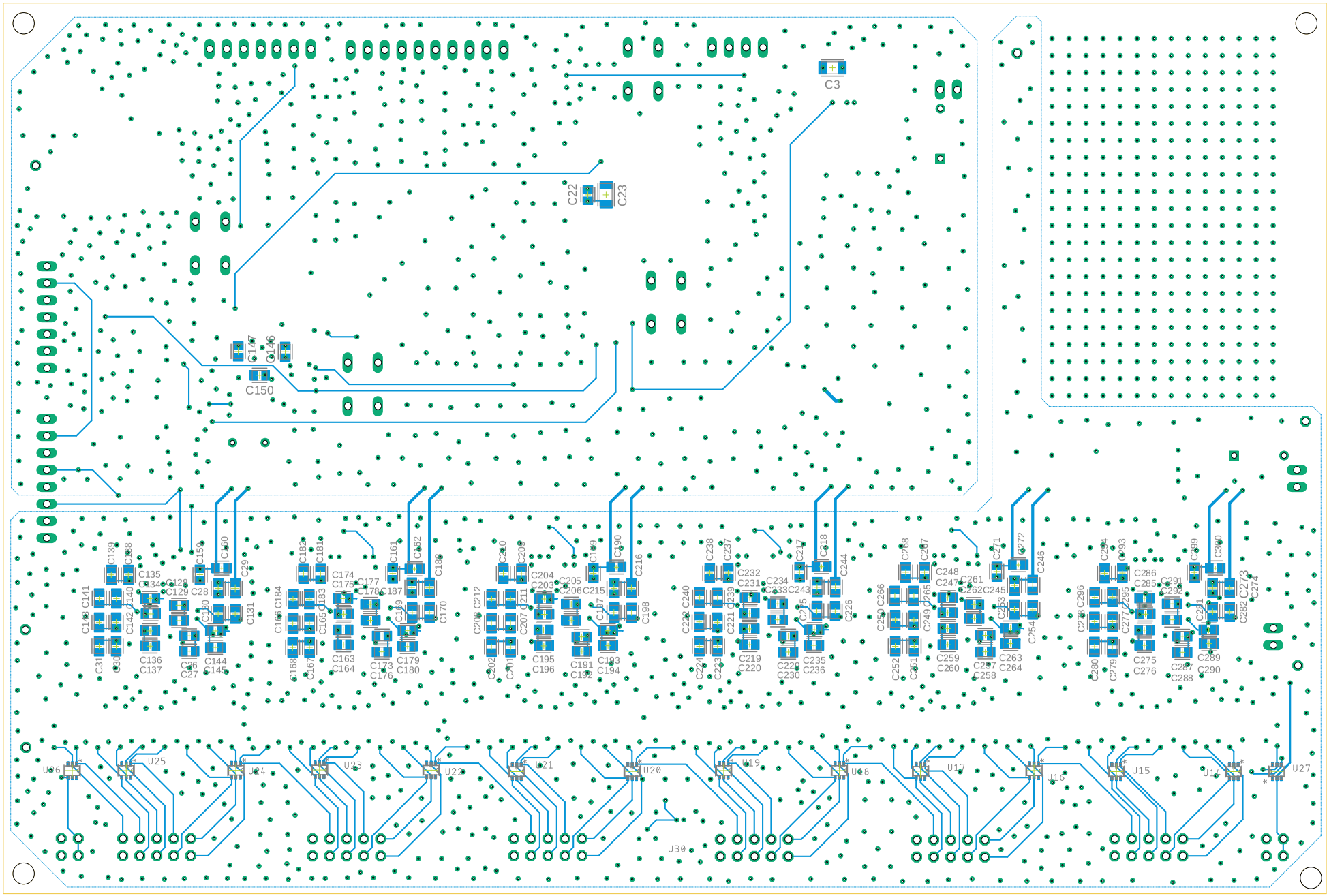
Table 1: Copper Fill Layer Stack


EDL-2023
TUE-06
AAYUSH RAJESH
ADITYA SRIRAM
SHIVAM PATEL
TUE-18
ANKITH R
ANUBHAV BHATLA
NITHISH KUMAR REDDY
ASSISTED BY:
MAHESHWAR MANGAT
ANKUR AGARWAL
PROF. SIDDHARTH TALLUR
PROF. LAXMEESHA SOMAPPA
ESP-BOOT
ESP-RST
PIC-BOOT
MCLR/RESET
XTAL1
U1
U7
LIS3DHHTRQFN
U28
U13
U12
U11
U10
U9
U8
U30
BAT85
MISO1
MOSI1
MISO2
MOSI2
AGND
DGND


Designator



26-03-2023 09:53 f=1.38 C:\Users\AAYUSH~1\AppData\Local\Temp\Neutron\ElectronFileOutput\11924\brd-3ad7fbcb-1994-42f9-9d68-df653e6485a3\BCI_21_March_2023 v11.brd


C3
C22
C23
C150
C147
C146
U26
U25
U24
U23
U22
U21
U20
U19
U18
U17
U16
U15
U14
U27
U30

Top Layer

Almost all of our components are housed on the top layer. The lower half of the top layer is reserved for the Analog devices, such as the electrode connections, analog front-end filters, and the ADCs. Therefore, we have a copper fill of AGND in this region, which extends near the power regulators as well.

The top half of the layer has our digital devices, namely the microcontroller, WiFi chip, SD card, and accelerometer. Therefore, we have a DGND copper fill in this region. One notable connection is the accelerometer, which has pads under the device and is therefore applied with a stop mask surrounding it.

All our connectors - electrode connectors, battery connectors, test points, and headers for programming and debugging - are placed at the edge of the board. We had an empty region to the top-left of our board, and therefore made it a prototyping area by placing through-hole vias in the region. The Analog and Digital fills are connected by placing Ferrite beads along the region between their copper fills.

Layer 2

This layer is mainly a power layer and is also used minimally for routing (as can be seen from the layout). As mentioned in Table 1, the lower half has an AVDD copper fill, and the upper half has a DVDD fill. While the WiFi chip uses the same value as DVDD as the other digital components, it requires a higher power rating and therefore has a copper fill of its own.

Layer 3

This layer is also predominantly a power layer and has fewer routes than Layer 2. The lower half has an AVSS fill, and since we did not require this layer for the power requirements of the digital half, we gave it a DGND fill.

Bottom Layer

The bottom layer houses the ESD protection diodes and the majority of the decoupling capacitors of the devices. This is to ensure that the distance between the capacitor and the corresponding pin of the device is as short as possible, as the connection between them can be achieved using a very short trace from the pin followed by a through-hole via to the pad of the capacitor. As seen from the layout (which has been mirrored), the AGND and DGND copper fills of the bottom layer mirror those of the top layer for signal stability. The top and bottom layer fills are connected using stitching vias.

Test Results

Due to the large number of components being used, we have divided the testing process into the following sections:

- Interfacing the ESP-8266 Wi-Fi module
- Single ADC interfacing using Arduino
- Interfacing Daisy-chained pair of ADCs using Arduino
- SPI interfacing for the PIC32 Micro-controller
- Interfacing Daisy-chained ADCs with PIC32

Interfacing the Wi-Fi Module

We configured the ESP8266 Wi-Fi chip as an access point using Arduino and ESP8266WiFi library. This will allow other devices like smartphones and laptops to connect to. To see if the network is established or not, open Wi-Fi settings and look for the network named "ESP8266 Access Point" (Figure 7) and connect to it by using the password we set in the Arduino code. To check the connection, we can ping the IP address 192.168.4.1 (default IP address of ESP Access point) using the laptop terminal and see if the chip responds to the pings.

Single ADC interfacing

The ADS1194 chip was connected to a breadboard using the Schematic connections for the last ADC. The SPI pins ($\overline{\texttt{CS}}$, `MISO`, `MOSI`, `SCLK`) were connected to the Arduino pins 10, 11, 12, 13, respectively. `RESET` was connected to pin 5 and $\overline{\texttt{DRDY}}$ to the pin 6 of the Arduino. The Arduino also provides the 3.3V and GND supplies to ADC. `CLKSEL` was tied to `HIGH` in order to use the internal 2MHz clock for the ADC. The RLD pins were left floating as they are only meant for ECG applications. The remaining pins were connected to the required signals as mentioned in the schematic.

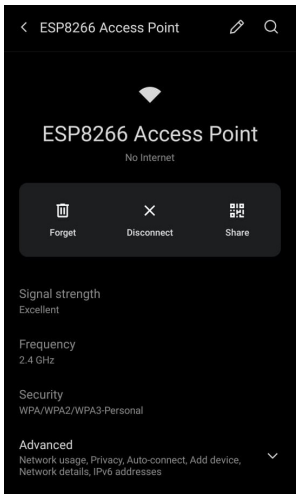


Figure 7: Successful connection to ESP8266 Access point network

A basic SPI interfacing code was uploaded on the Arduino which calls the `SPI.begin()` function and initializes the SPI to the required settings. The initial startup sequence was followed as mentioned in the datasheet and the ADC Config registers were written to initialize the test signals. The ADC registers were synced successfully and printed on the Serial monitor, as shown in Figure 8.

Register 0x00 (ID) has the value 10110100. Bits[7:5] denote that the device belongs to the ADS119X family. Bit 2 should be `HIGH` for this family and the Bits[1:0] are used for channel number identification. Bits[1:0] = 00 denotes that the connected device is an ADS1194.

Register 0x01 (CONFIG1) has the value 00000100. Bit 6 ($\overline{\texttt{DAISY\_EN}}$) is reset to `LOW` by default. Bit 5 (`CLK_EN`) is reset to `LOW` to disable oscillator clock output. Bits[2:0] are 100 by default which sets the sampling rate to 500SPS.

Register 0x02 (CONFIG2) has the value 00110000. Bit 4 (`INT_TEST`) is set to `HIGH` to generate the internal test signals. Bit 2 is used to set the test signal amplitude and is reset by default. Bits[1:0] are used to set the test signal frequency, which generates an $\approx$ 0.95Hz test signal.

Register 0x03 (CONFIG3) has the value 11000000. Bit 7 ($\overline{\texttt{PD\_REFBUF}}$) enables the internal reference buffer. Bit 5 is reset to `LOW` by default and VREFP is set to 2.4V. The remaining bits control the RLD circuitry and are reset to 0 by default.

Registers 0x05 to 0x08 are used to control Channel settings and have an identical value of 00000101. Setting Bits[2:0] to 101 sets the input for Channels 1-4 as the test signal.

```
14:11:56.395 -> ADS119X Conected
14:11:56.395 -> REGISTERS:
14:11:56.395 ->  0x0: 10110100
14:11:56.395 ->  0x1: 00000100
14:11:56.395 ->  0x2: 00110000
14:11:56.395 ->  0x3: 11000000
14:11:56.395 ->  0x4: 00000000
14:11:56.395 ->  0x5: 00000101
14:11:56.395 ->  0x6: 00000101
14:11:56.395 ->  0x7: 00000101
14:11:56.395 ->  0x8: 00000101
14:11:56.395 ->  0x9: 00000000
14:11:56.395 ->  0xA: 00000000
14:11:56.395 ->  0xB: 00000000
14:11:56.395 ->  0xC: 00000000
14:11:56.395 ->  0xD: 00000000
14:11:56.395 ->  0xE: 00000000
14:11:56.395 ->  0xF: 00000000
14:11:56.395 -> 0x10: 00000000
14:11:56.395 -> 0x11: 00000000
14:11:56.395 -> 0x12: 00000000
14:11:56.395 -> 0x13: 00000000
14:11:56.395 -> 0x14: 11111111
14:11:56.395 -> 0x15: 00000000
14:11:56.395 -> 0x16: 00000000
14:11:56.395 -> 0x17: 00000000
14:11:56.395 -> 0x18: 00000000
14:11:56.395 -> 0x19: 00000000
14:11:56.395 -> Num Channels: 4
```

Figure 8: Registers values for generating test signals

The converted digital values from these 4 channels are then sent to the Serial Monitor. We observe a 0.97Hz square wave on all 4 channels, as shown in Figure 9.

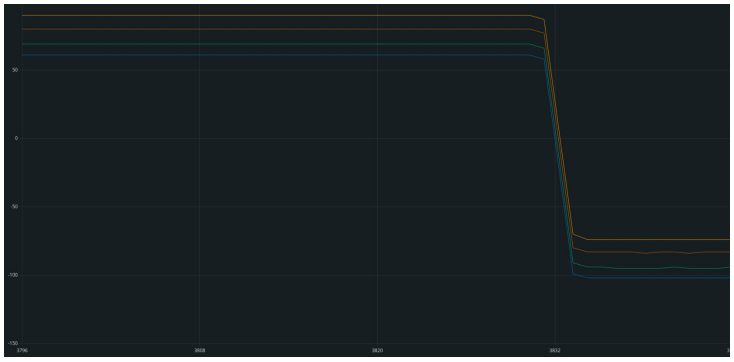


Figure 9: Test Signal observed on the Serial Plotter

After the test signals have been observed, the register values were written, as shown in Figure 10 to switch the channels to the Normal operation Mode, the Gain is set to 1 and the Data Rate is set to 500SPS. Registers 0x05 to 0x08 now have a value of 00010000. Bits[6:4] are used to set the PGA gain, which in this case is set to 1. Setting Bits[2:0] to 000 sets the channel to use Electrode inputs.

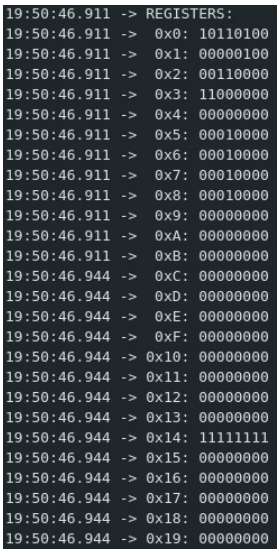


Figure 10: Registers values for normal operation

Since this current set-up has been connected to a Breadboard (as seen in Figure 11), it is prone to vulnerabilities and observations may change with every iteration. This was especially evident when we had a few failed attempts at connecting with the ADC after our initial successful connection, having us resort to testing individual SPI signals and viewing them on an oscilloscope (Drive Link). Since attaching another ADC and daisy-chaining will increase the complexity of the current circuit, we are planning to migrate to a solderable test bed.

SPI Interfacing for Microcontroller

The microcontroller our design uses belongs to the PIC32 family but has not arrived yet. In order to gain familiarity with this kind of microcontroller, and iron out any issues with our developing codebase, we have begun testing with a microcontroller module belonging to the same family - the PIC32MX534 clicker board. We have begun debugging issues with operating on this board, and we shall begin migrating our current Arduino testing set-up to this board in order to interface with our test ADCs. Since the board belongs to the same family, we do not expect many changes to be required in our final codebase after we complete testing with this module.

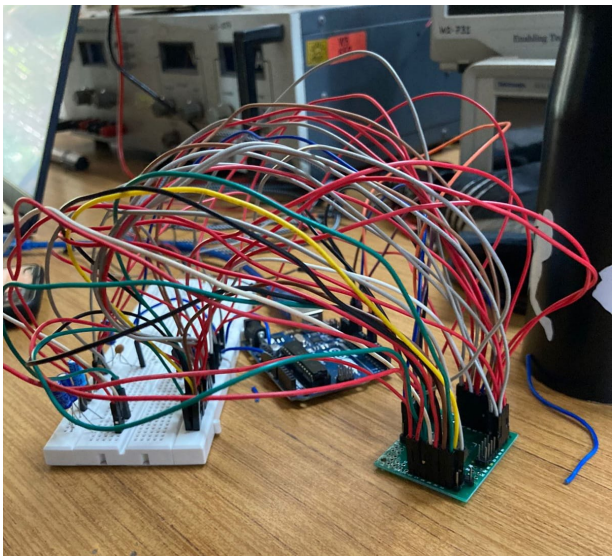


Figure 11: Breadboard testing setup interfacing Arduino with ADC

Progress made so far

The following table shows the order in which we are planning to conduct testing and the progress made in each step so far. Wi-Fi Interfacing has been successfully completed. We have obtained some results from interfacing a single ADC and are currently migrating the breadboard circuitry to a prototyping board. We are also writing a basic code in MPLAB to interface with the PIC32 clicker board to do some firmware development until our required PIC32 Microcontroller arrives. We plan to move to Daisy-chain interfacing once we are convinced with the results obtained from interfacing a single ADC. Once we have completed the above testing steps and our microcontroller arrives, we plan to migrate to the PIC32 Microcontroller and start firmware development for interfacing all the components simultaneously.

Subsystem	Testing Process	Progress
Wi-Fi	Wi-Fi Interfacing	100%
ADC	Single ADC Interfacing	30%
	Migrating to Prototyping Board	50%
	Daisy Chain Interfacing	Pending
Microcontroller	PIC32 Interfacing	20%
	Migrating to PIC32	Pending

Table 2: Progress made so far

Next Steps

The current tasks at hand are mainly aimed at testing of the ADCs (including daisy chaining), and tuning our codebase implementation for our particular PIC32x family microprocessor. To delineate our tasks -

- **Implementing a Prototyping Board** - We need to complete the integration of the ADC to the prototyping board, which will help us save time from debugging wires, and also ease the connections when we daisy chain two ADCs.
- **Testing Daisy Chaining** - We need to test out the daisy chaining of two ADCs to understand exactly how the data transfer occurs. Our major focus shall be on the data format for slave-slave communication and synchronization between two ADCs based on CLK signal provided by a single ADC.
- **PIC32 Interfacing** - Parallelly, We will be working on interfacing and programming the PIC32 Clicker board using PICKit and ironing out bugs in our codebase.
- **Migrating Testing to PIC32** - Upon completion of above tasks, we shall be migrating our testing from the Arduino UNO-3 to the Clicker board, and all the above tests will be re-run to ensure code compatibility with PIC family microcontrollers.

Valuable insights drawn from the above tests will help us finalize our code framework, which will facilitate a smooth transition to implementation on the PCB upon its arrival. The PCB Design and Hardware sub-teams have been shifted to focus on the testing process and will be working closely with the Firmware sub-team in order to gather the necessary details for finalizing the codebase.

The agenda for the week of 27^{th} March - 2^{nd} April is to complete designing the solderable testbed and testing and debugging ADC SPI connections on it. Parallely to this, we shall also look to complete interfacing on the PIC32 Clicker board. We shall also look to test data format in Daisy Chain transmission.

References

1. OpenBCI Documentation
2. https://arxiv.org/ftp/arxiv/papers/1808/1808.03711.pdf
3. ADS1194 Datasheet
4. PIC32MX250F128B/D Datasheet
5. ESP8266EX Datasheet
6. AP2112 Datasheet
7. LM2664 Datasheet
8. TPS72325 Datasheet
9. TLV700 Datasheet
10. LIS3DHHTR Datasheet
11. TPD4E1B06DCKR Datasheet