EE344 - Electronic Design Lab
OpenBCI based EEG Acquisition System
Final Report

Group 1 : TUE-29
Members : Madan Y N (200070040)
Kushal Pawaskar (20D070059)
Vaibhav Kumar (20D070087)

Group 2 : MON-30
Members : Gowtham S (20D070031)
Saarthak Mittal (20D070066)
Utkarsh Jindal (200070086)

Guided by :
Prof. Siddharth Tallur
Prof. Laxmeesha Somappa

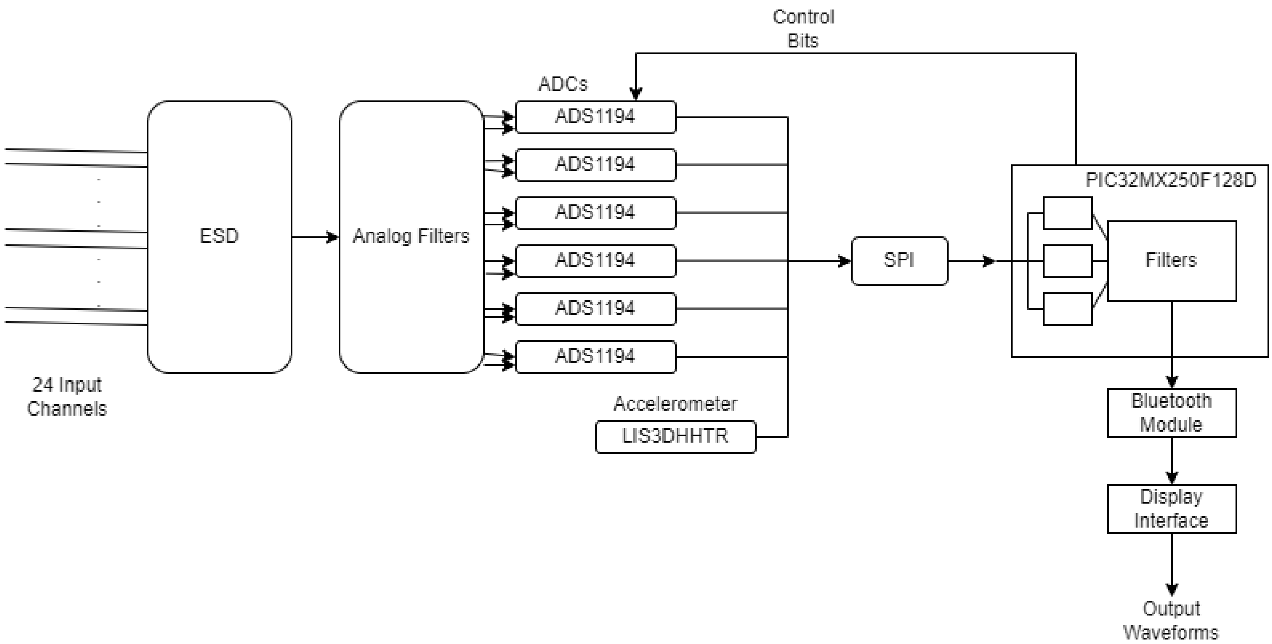
Contents

1 Problem Statement

The aim of the project is to develop an ELectroencephalogram(EEG) Acquisition System that is capable of monitoring brain activity through 24 different channels simultaneously. The system should convey the captured data in real-time to a computer over Bluetooth and the receiver should plot waveforms of all the channels. The design is inspired by the OpenBCI Cyton/Ganglion biosensing board, which is a well-known and widely used EEG system.

2 Design

2.1 Block Diagram



Explanation

The whole flowgraph can be divided into two broad sections which are namely data acquisition and data communication. The data acquisition consists of input channels which are 24 in number, ESD protection, low-pass filters to filter high frequency noise (EEG signals are typically in the range of few tens of hertz), ADS1194 which is a 16 bit ADC and LIS3DHHTR, an accelerometer to track head movements. Data communication consists of HC-05 bluetooth module to serially communicate data and a receiver end software to receive and plot waveforms in real-time.

Principle of Operation

The entire circuit can be classified into two subsystems based on two primary functions: data acquisition and processing, and data transmission. The principle of operation of each subsystem is as below.

- **Data Acquisition**

  The data acquisition subsystem of our circuit consists of six $ADS1194$ Analog-to-Digital Converters, $LIS3DHHTR$ Accelerometer and SD Card port. All these ICs are connected to $SPI0$ module of the microcontroller and are operated through time division multiple access.

Setup
This is the first stage in the operation of our circuit. Initially we will boot the micro-controller with our firmware through the $UART1$ module. After boot up, communication interfaces $SPI0$, $UART0$ ports are initialized. Next up, we set the programmable gains of all the six ADCs to 12 (the maximum allowed gain).

Update Channel Data
Default mode is the normal mode of operation. The chip select lines of the all the six ADCs are sequentially drawn low one at a time and a packet of data corresponding to one channel is received from each ADC. This way data is received from all the 24 channels in sequence; the raw data is processed and stored in an array.

Update Accelerometer Data
Similar to the previous stage, in this stage the chip select line corresponding to the accelerometer is drawn low. Data is read, processed and stored in an array.

Processing
In this stage input characters is received from PC through $UART1$ and according to the input the operation of the device is decided. The mode of operation of the board, from how many and from which all channels is the data needed to be collected, and many other control related decisions can be made in this stage.

- **Data Transmission**

HC-05 BLuetooth Classic Module
HC-05 provides fast and reliable data transmission. It has on-board 3.3 V voltage regulator so it can work with 5V as well.We are going to use UART between PIC μC and HC-05. It has few important pins as follows:

1. Enable: this pin is used to toggle between data and command mode.
2. Vcc : +5V power supply
3. Ground : 0V connection
4. Tx : transmits serial data, everything received via Bluetooth will be given out by this pin as serial data.
5. Rx : receives serial data, serial data given to this pin will be broadcasted via bluetooth.

Packet Format
Each packet contains a header byte, 24 channel data and accelerometer data, and a footer byte. The description is as follows:

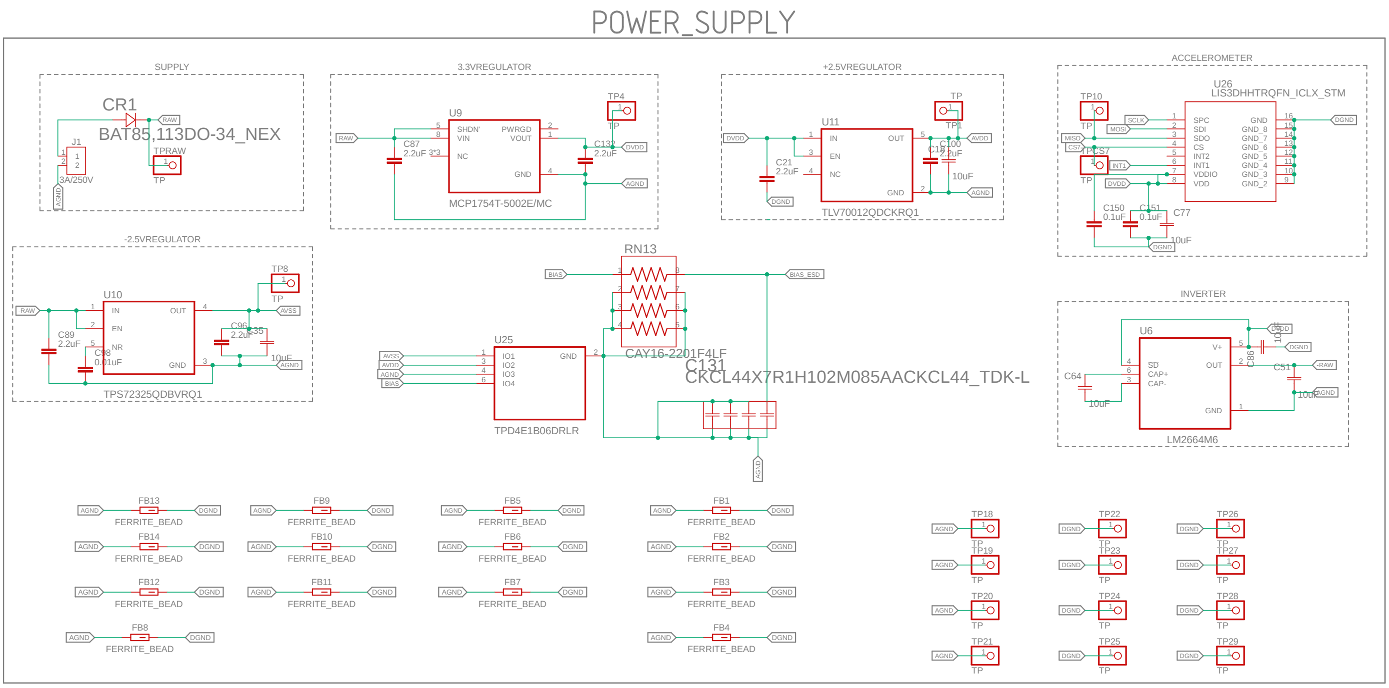
Byte Range	Data Contained
1	start byte
2-3	channel 1 data
⋮	⋮
48-49	channel 24 data
50-51	accelerometer X data
52-53	accelerometer Y data
54-55	accelerometer Z data
56	stop byte

Processing at PC side

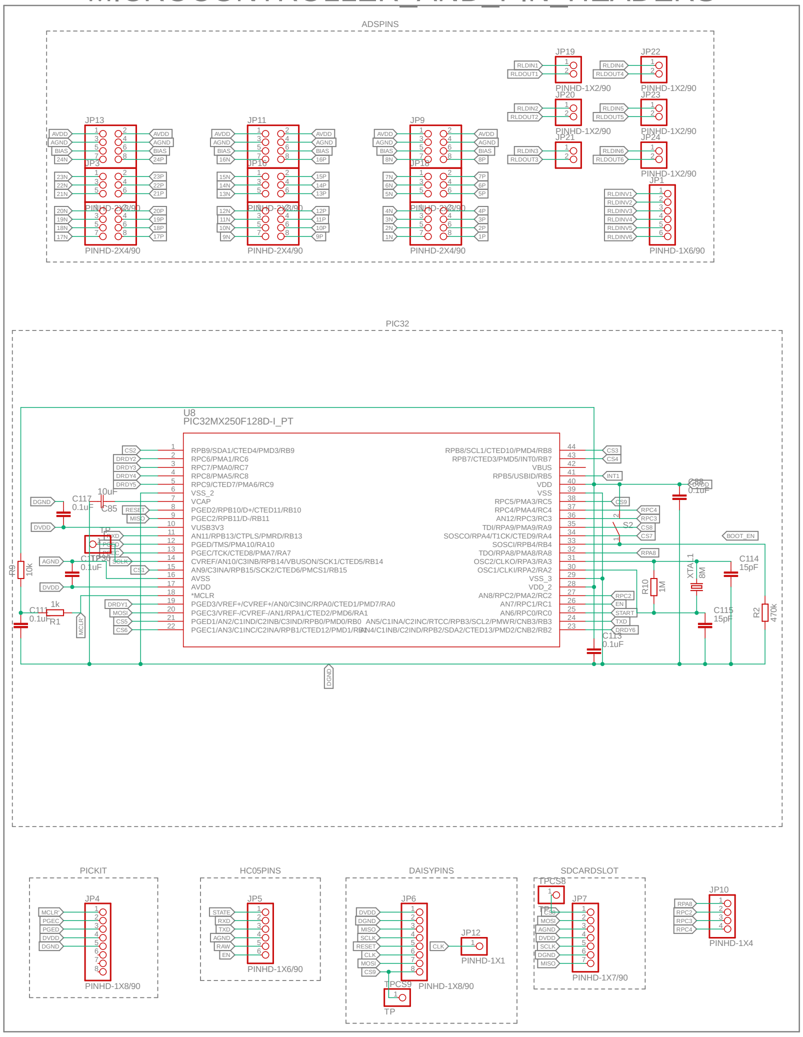
1. We intended to show live time series of 24 channel EEG data and locally record the data in csv format for later analysis.
2. We intended to show plots of movement of head in 3 directions using accelerometer data and record it as well.

2.2 Design Schematic

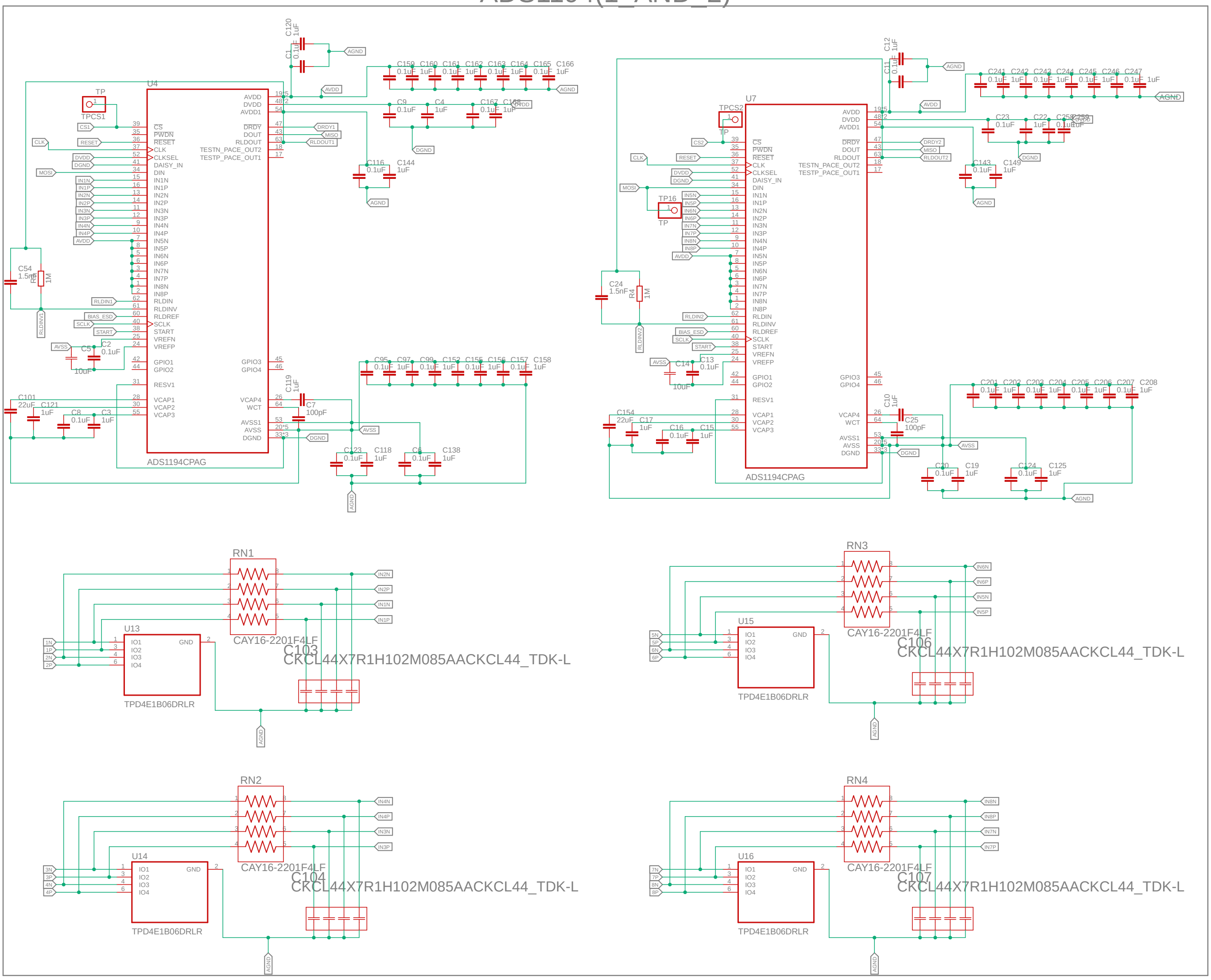
Final PCB



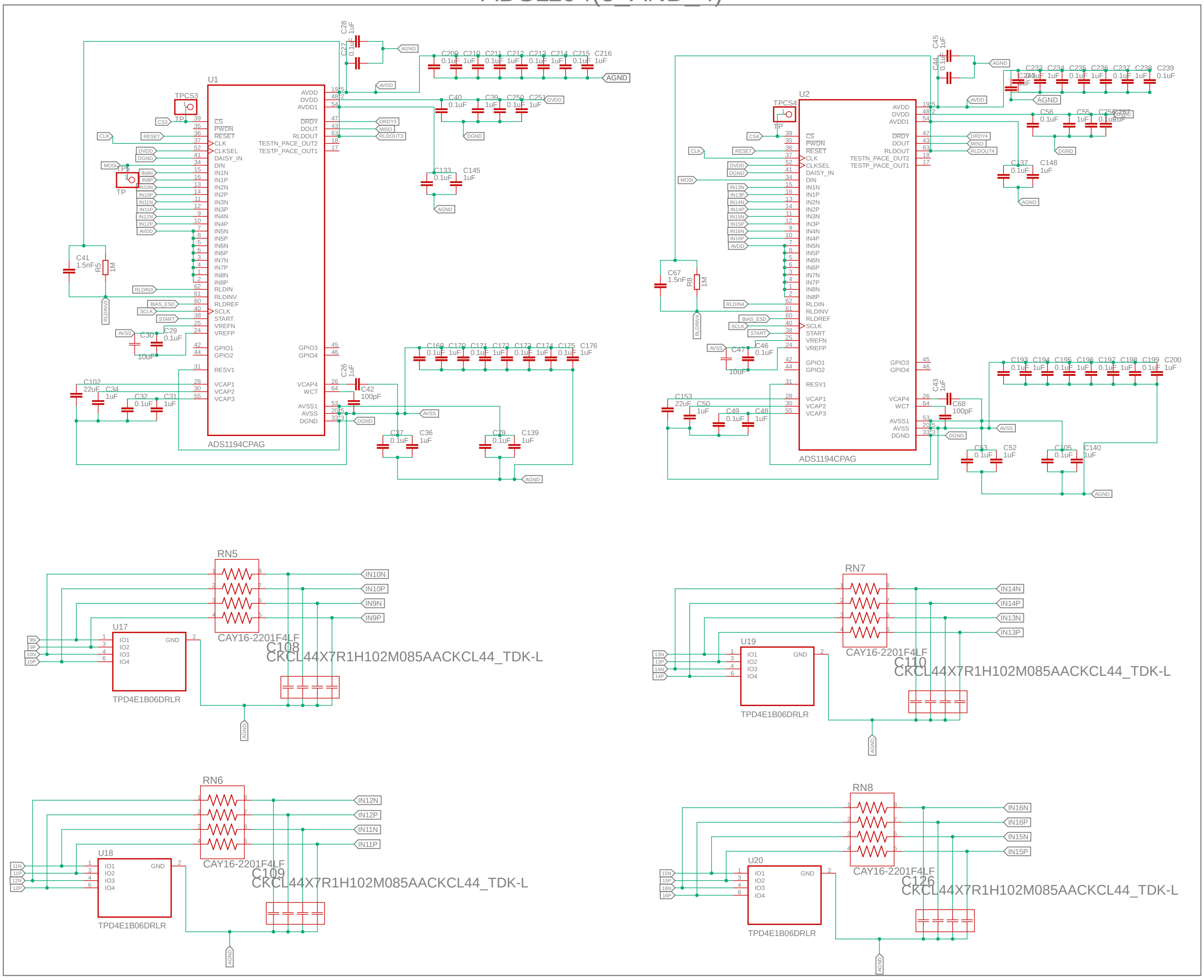
MICROCONTROLLER_AND_PIN_HEADERS

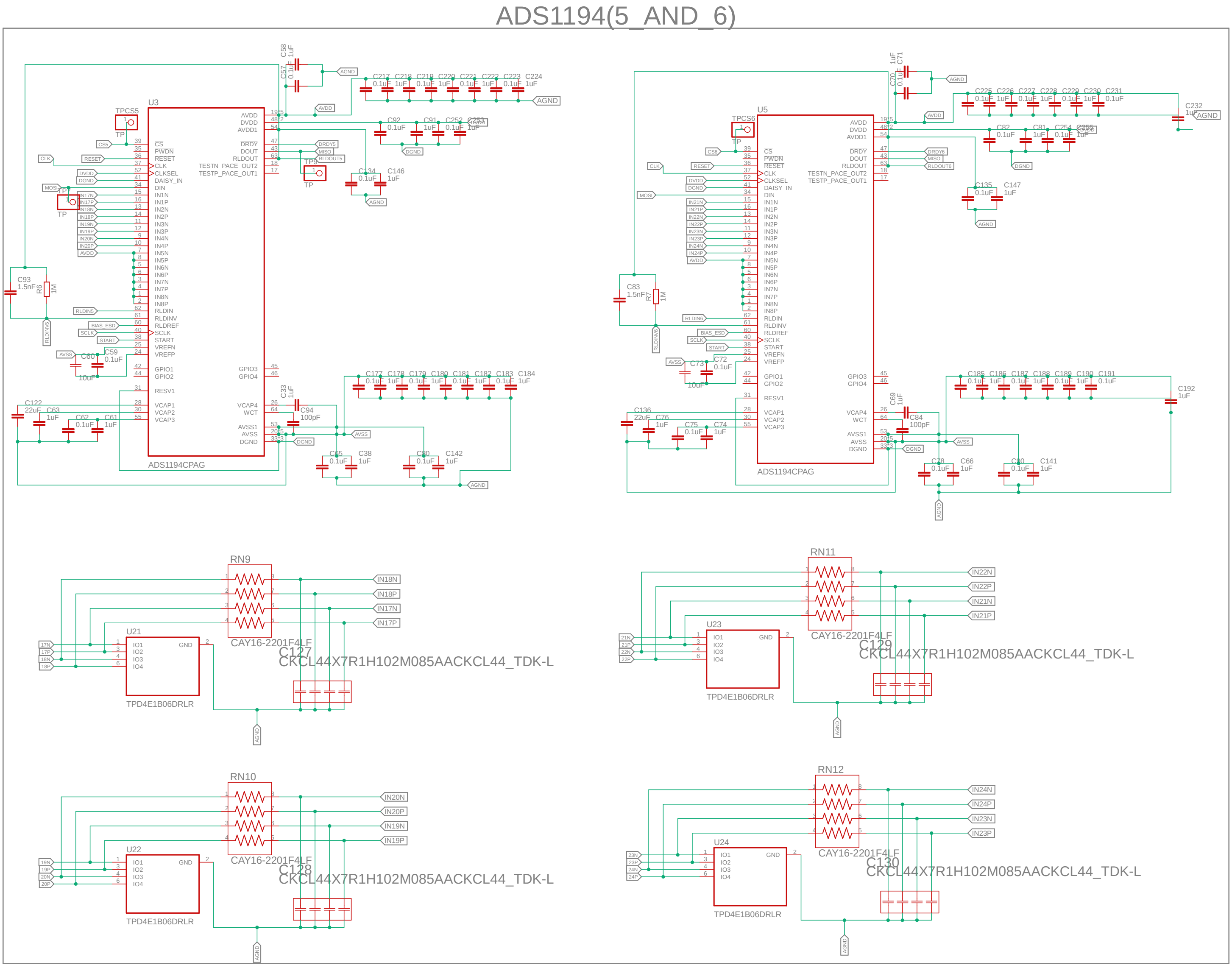


ADS1194(1_AND_2)

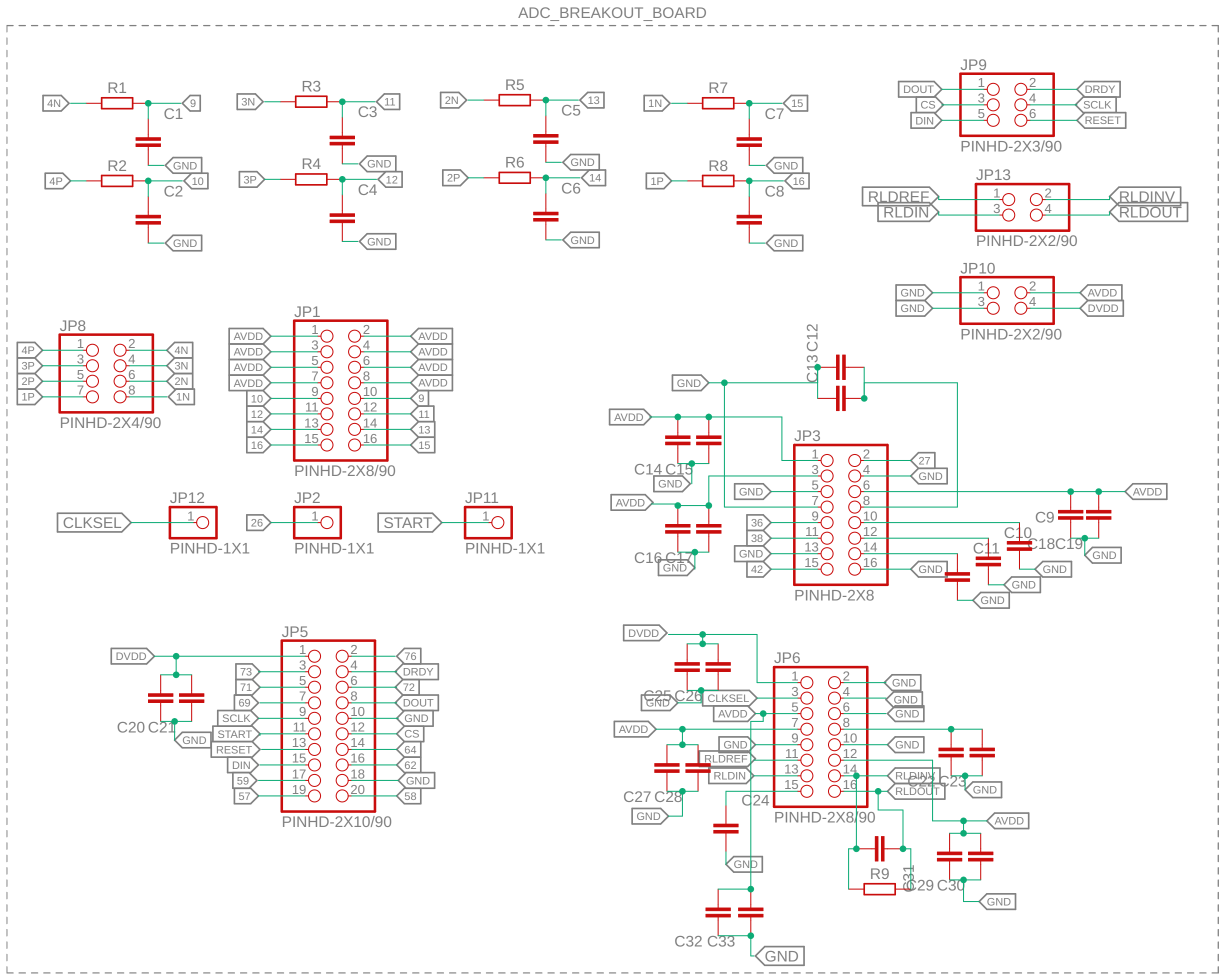


ADS1194(3_AND_4)




ADS1194(5_AND_6)
U3
ADS1194CPAG
U5
ADS1194CPAG
RN9
RN10
RN11
RN12
CAY16-2201F4LF
U21
U22
U23
U24
TPD4E1B06DRLR
C127
C128
C129
C130
CKCL44X7R1H102M085AACKCL44_TDK-L

Breakout PCB

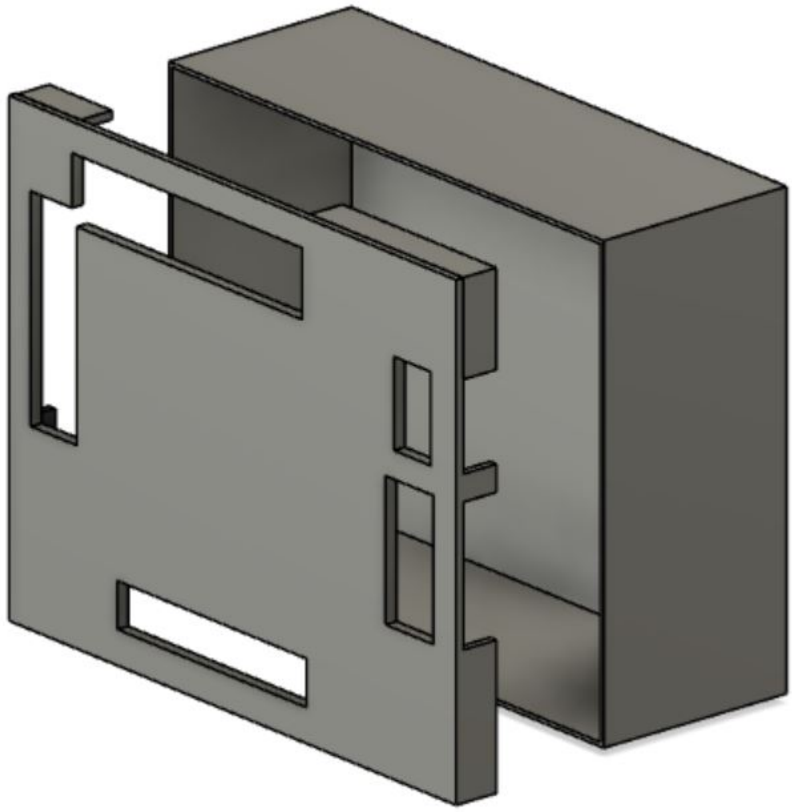


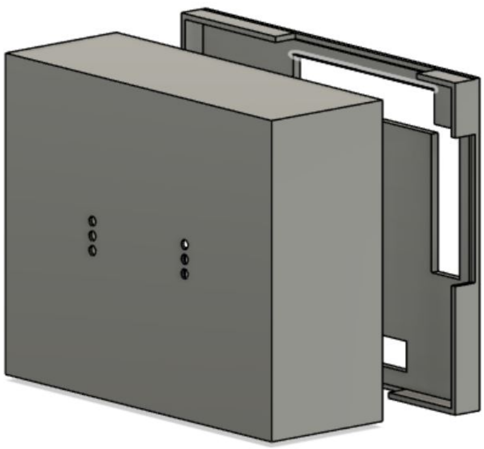
2.3 Bill of Materials

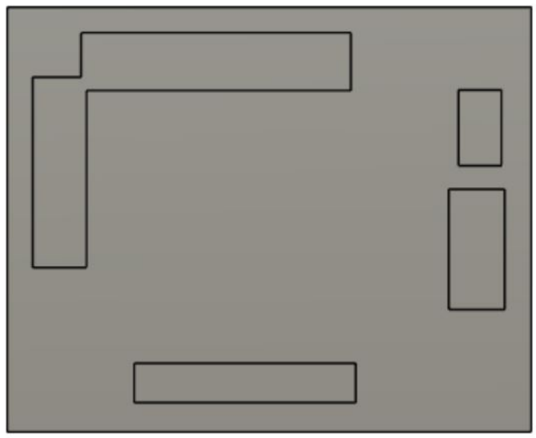
Item Description	**Component**	**Rate(in Rs.)**	**Quantity**	**Amount(in Rs.)**
ADC	ADS1194	1135.53	6	6813.18
Microcontroller	PIC32MX250F128D	463.17	1	463.17
Accelerometer	LIS3DHHTR	1000	1	1000
Transient Voltage Suppressor	TPD4E1B06DRLR	35.82	20	716.4
Voltage Regulator($+3V$)	MCP1754T-5002E/MC	66.75	2	133.5
Voltage Regulator ($-2.5V$)	TPS72325QDBVRQ1	301.99	1	301.99
Voltage Regulator ($+2.5V$)	TLV70012QDCKRQ1	64.31	1	64.31
Voltage Inverter	LM2664M6X/NOPB	74.89	1	74.89
$0.1\mu F$ Capacitor	04023C104KAT2A-62	30.12	18	542.16
$1\mu F$ Capacitor	C1206C105K3NACAUTO	43.14	16	690.24
10 μF Capacitor	C2012X7S1E106K125AE	56.17	6	337.02
18 pF Capacitor	CBR04C180J5GACAUTO	17.09	4	68.36
1 $M\Omega$ Resistor	RN73H2BTTD1004B25	58.61	2	117.22
470 $k\Omega$ Resistor	RN73H2ATTD4703B25	86.28	3	258.84
$1K\Omega$ Resistor	RN73H2BTTD1001B10	90.35	2	180.7
Capacitor Array	CKCL44X7R1H102M085AA	29.30	17	498.1
$2.2k\Omega$ Resistor Array	CAY16-2201F4LF	8.29	17	140.93

Total Amount : Rs. 13,726.55

2.4 CAD Models







3 Results and Observations

3.1 Data Acquisition

Data acquisition from ADC has two parts: firstly, interfacing the ADC with microcontroller and reading the default register settings of the ADC and secondly, giving analog input signals to the input lines of the ADC and obtaining the converted digital values. It should be noted second part cannot proceed without completing the first part. We did four iterations of the first part and they are as follows,

- **Iteration 1** — Breadboard circuit of ADC with Arduino development board
  - AVDD - 5V from external source
  - DVDD - 3.3V from external source

  The setup and the ADC supply connections made are given in the below images

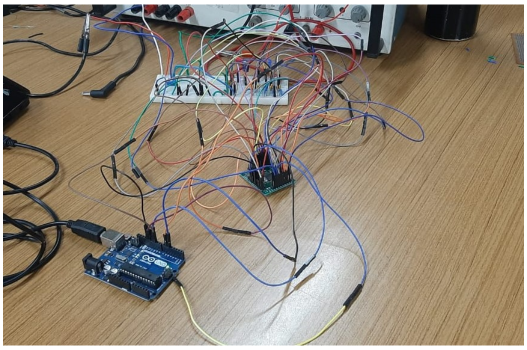
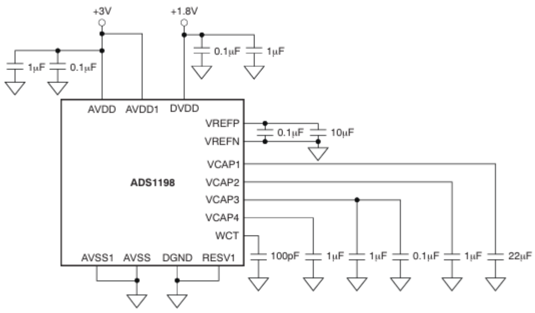



Figure 1: Setup (left) and ADC supply connections(right)

  Result: The ADC gave 0x00 as reading of all the registers which implies that the ADC power-on hasn't been done properly.

  Possible Reason: Arduino development board has a logic high level of 5V but the ADC has digital high of 3.3V and this might have damaged the IC and consequently resulted in failure of the test.

- **Iteration 2** — Breakout PCB with PIC32MX clicker board
  - AVDD - 5V from clicker board
  - DVDD - 3.3V from clicker board

  The ADC supply connections made are same as those of iteration 1. The digital high of the clicker board is 3.3V which matches with logic high of the ADC.

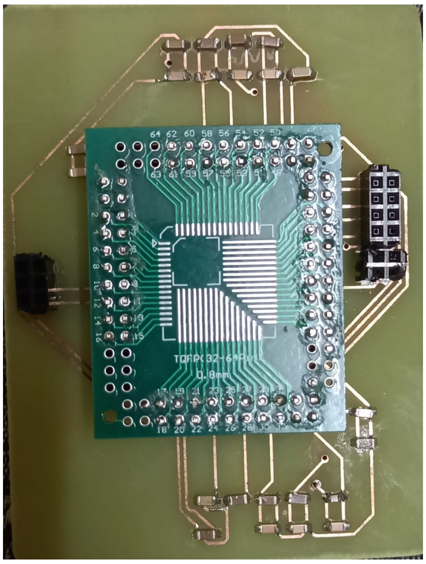


Figure 2: Breakout PCB with ADC setup

Result: The ADC gave 0x00 as reading of all the registers which implies that the ADC power-on hasn't been done properly.

Reason: Error was found in some of the decoupling capacitor connections which caused the effective voltage at the pin lower than the required voltage.

- **Iteration 3** — Breadboard circuit of ADC with clicker development board
  - AVDD - 5V from clicker board
  - DVDD - 3.3V from clicker board

  The ADC supply connections made are same as those of iteration 1 & 2. The digital high of the clicker board is 3.3V which matches with logic high of the ADC.

  Result: The ADC gave 0x00 as reading of all the registers which implies that the ADC power-on hasn't been done properly.

  Possible Reason: In this iteration of the testing, we used the same ADC chip that was used in iteration 2, which may have been damaged while doing iteration 2 .

- **Iteration 4** — New breakout PCB board The mistakes in the previous iteration on breakout pcb were rectified and a new breakout PCB was designed.

  Result: The testing could not be done since PCB could not be printed in time for the submission.

  Reason: It was too late when we got to know that PCBLab was under maintenance and outside vendors could not deliver the PCB in time.

3.2 Data Communication

The objective is to transmit 24-channel data (16-bit each) from μC to PC. We used bluetooth for transmitting data thorough UART. We used HC05 bluetooth module for this purpose. The main tasks in data communication are transmitting data to PC using HC05, plotting real-time channel data, and storing locally in csv file for any other purpose.

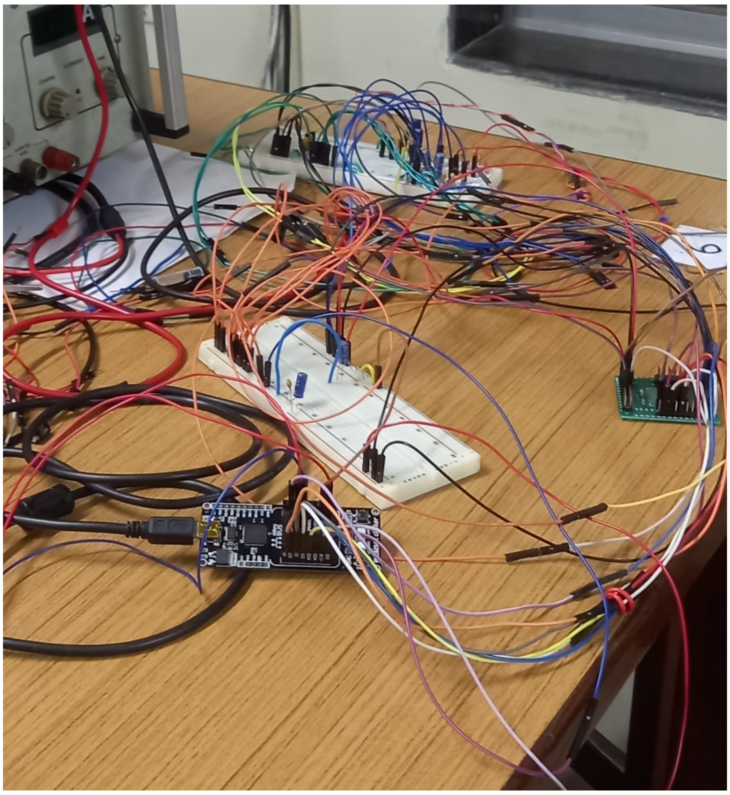
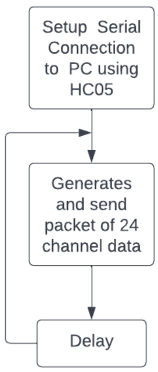


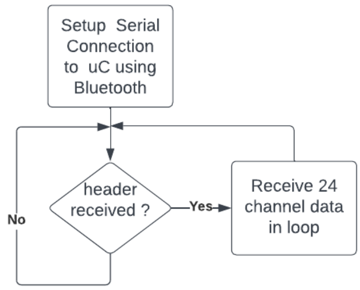
Figure 3: Breadboard circuit setup for ADC testing using clicker board

Block diagrams

- **Data transmission at microcontroller :** Since we were not able to finish ADC part so here we generate uniformly random 24 channels data (16 bit each) for transmission. We used UART for serial connection. We have added a delay in data generation to simulate the behaviour of ADC because it digitise input upto a fixed rate.The block diagram for code



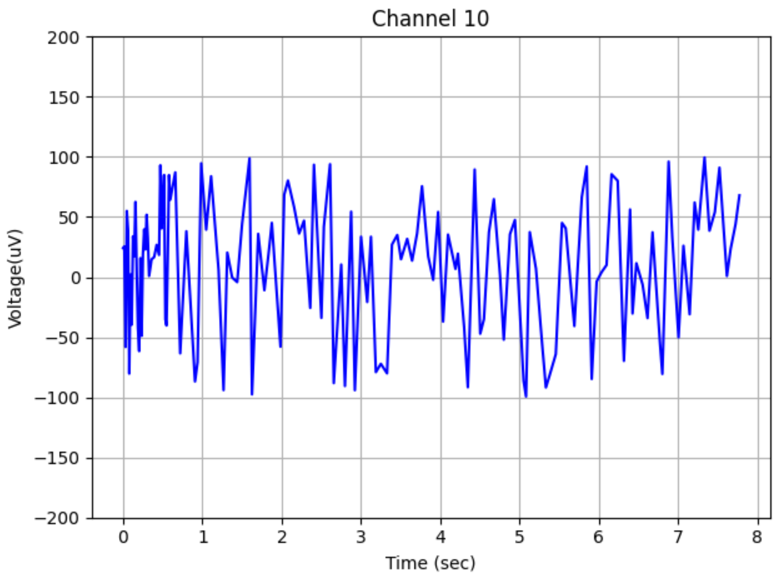
- **Data reception at PC:** We used PySerial Python module for serial connection to HC05.We run a python script on PC which continuously looks for packet header and then collects 24 channel data ,write all channels data to csv file. 16 bit channel data is appropriately converted to (typical) range ($-100\mu V$ to $100\mu V$) using gain factor as per ADC resolution.



We did two iteration for bluetooth and the results are as follows :

- **Iteration 1:** We used arduino to send the channel data through UART using HC05 and used python to receive channel data as described in block diagram.

  Result: We were able to successfully transmit and receive data through using bluetooth, we stored it locally in csv file but we were not able to plot the data in real time because data is received at fast rate (1 packet of 24 channel data every 50-60ms) and real time updation of plot is causing PC to froze. Hence, we stored the data and plotted it locally later after complete reception.

  Video Demonstration :
  `https://drive.google.com/drive/folders/1LYR-3z1IbRmSmTMcULbhplSOKwMd8xXk?usp=sharing`

- **Iteration 2:** In this iteration, we did transmission of data with clicker (development) board since our pcb is not ready and ADC part is still not completed. The data receiving and plotting is same as iteration 1.

  Result: This iteration was not fully successful. The data transmission from clicker board using uart with bluetooth was happening but the data received was not in correct encoding. We tried various different decoding format but couldn't find the solution. We don't know exact reason behind the problem but our guess is either encoding issue or pull up register issue.

4 Conclusions and Future Work

The aim of the project to build a 24-channel EEG acquisition system could not be achieved in the stipulated time. We completed a compact standalone PCB for the whole 24-channel system. We also demonstrated transceiving data over bluetooth. Future work would start with completing iteration 4 of the ADC testing and then move to channel data acquisition by a single ADC and finally by all the six ADCs as in the final design. One more improvement on the design may be, instead of designing a single PCB with all 24 channels, we can make a modular design which would allow customization of number of channels as 8 or 16 or 24.