Kshamta: A Disability Aid

EC211P Electronics Circuit Lab

Contributors:
- BT2024003 Shaivi Nandi
- BT2024051 Ruthvik CSS
- BT2024262 Navya Balaji
- BT2024264 Utkarsh Goyal
- BT2024268 Vedant Mundada
- IMT2024079 Krisha Dhokariya

Date: 20th November 2025

Project objective:

Kshamta is a multi-purpose disability aid, designed to assist people with hearing disabilities as well as visual disabilities. We aim to bridge the gap in communication that is faced by them. Kshamta consists of three major modules, each of which are designed to perform one communication task:

1. Speaking Module: this converts speech input into text and sends it to the disability aid modules
2. Sign Language Module: this parses text and converts into sign language on one mechanical hand, covering 35+ various ASL signs that can be done using one hand only.
3. Braille Module: this parses text into letters and converts those letters into braille; printing them one by one on the pad. It even has a learning tool embedded, which allows one to input letters to learn braille and communicate.

Deliverables:

As mentioned in our project proposal, the system consists of two individual components- a braille system and a hand- designed for users who are blind, deaf, or both. The entire setup is controlled through a speech-to-text module, which converts the speaker's voice commands into text. This text then serves as the unified input for both systems.

1. Speaking Module: consists of a USB microphone integrated with an NRF radio transceiver on a Raspberry Pi microcontroller. This takes speech input from the microphone, then converts it into text using a locally-run speech-to-text ML model. This is then transmitted to the other modules in the form of a 32 byte packet terminated by a null character. If the packet is too big, it is fragmented and sent in chunks.
2. Sign Language Module: This is a mechanical hand, formed by the usage of cardboard and popsicle sticks (consisting of the body); and springs (to emulate the joints). Each finger is controlled by a 180 degree servo motor, where the angle of the servo arm controls the extent of bend of the finger. There is also a base, consisted of a cardboard circle, as well as a stepper motor that emulates the motion of the elbow. There is a receiver sub-module constructed on an Arduino Nano, consisting of an NRF radio transceiver. This submodule receives the packet from the speaking module, parses the packet into text, and sends it to an Arduino Uno connected to it. The Arduino Uno then makes the appropriate signals to produce the sign intended.
3. Braille Module: This system consists of 6 outputs as seen in a braille script. These outputs are managed by 6 servos which are powered by 2 Arduinos for equal and less power intensive supply of current. We have used 6 limit switches in order to take in inputs from the user. There are two buttons controlling the modes of the system- one that switches mode between reader and printer and one that switches between modes within the pre-existing modes i.e in case of printer mode, we have serial input and nrf(connected to the arduino) input settings and in case of reader mode it is used as an enter button. E have implemented inter arduino communication by connecting the TX-RX pins of one arduino to analog pins of the other arduino. We also have an LCD display mainly used to see which words have been recognized, which characters are being printed and what inputs are being given by the user.

Software / Firmware Design

Raspberry Pi Software

The Raspberry Pi captures audio using a USB microphone and processes it with the **VOSK offline speech-to-text engine** at 16 kHz. The code uses `sounddevice` to collect PCM samples and feeds them into the KaldiRecognizer. When a complete phrase is recognized, the resulting text is encoded into UTF-8 and transmitted over the **NRF24L01** wireless module.

The Pi configures the NRF24L01 with:

- Channel **0x76**,
- Data rate **250 kbps**,
- Payload **32 bytes**,
- Pipe address `[E0 E0 F1 F1 E0]`,
- CE = GPIO25, CSN = CE0.

Text is split into fixed 32-byte payloads and padded with `0x00` bytes. A corrected `transmit_text()` function ensures proper byte formatting and retries on failed transmissions.

Robotic Hand Arduino Firmware

The Hand Arduino stays in listening mode on the same NRF address.
Upon receiving a 32-byte packet, it converts bytes to a string, checks if the word matches any predefined sign-language gestures, and executes the corresponding servo sequence across 5 SG90 servos. Gestures are defined as lists of target angles, and the firmware moves servos gradually to avoid large current spikes.

Braille Printer Arduino Firmware

The Braille module also listens on the same NRF pipe. Incoming text is processed **character-by-character**. Each character is mapped to a 6-bit Braille pattern. Six servos raise/lower dots, and two additional servos feed the paper. An LCD displays the received character, and a button is used for advancing in manual/learning mode.

Communication Protocol

Communication uses the **NRF24L01 2.4 GHz GFSK radio** with the following configuration:

- Address: `[E0 E0 F1 F1 E0]` (same for both receivers)
- Channel: **118 (0x76)**
- Data rate: **250 kbps**
- Payload: **fixed 32 bytes**
- CRC & Auto-ACK: **enabled**

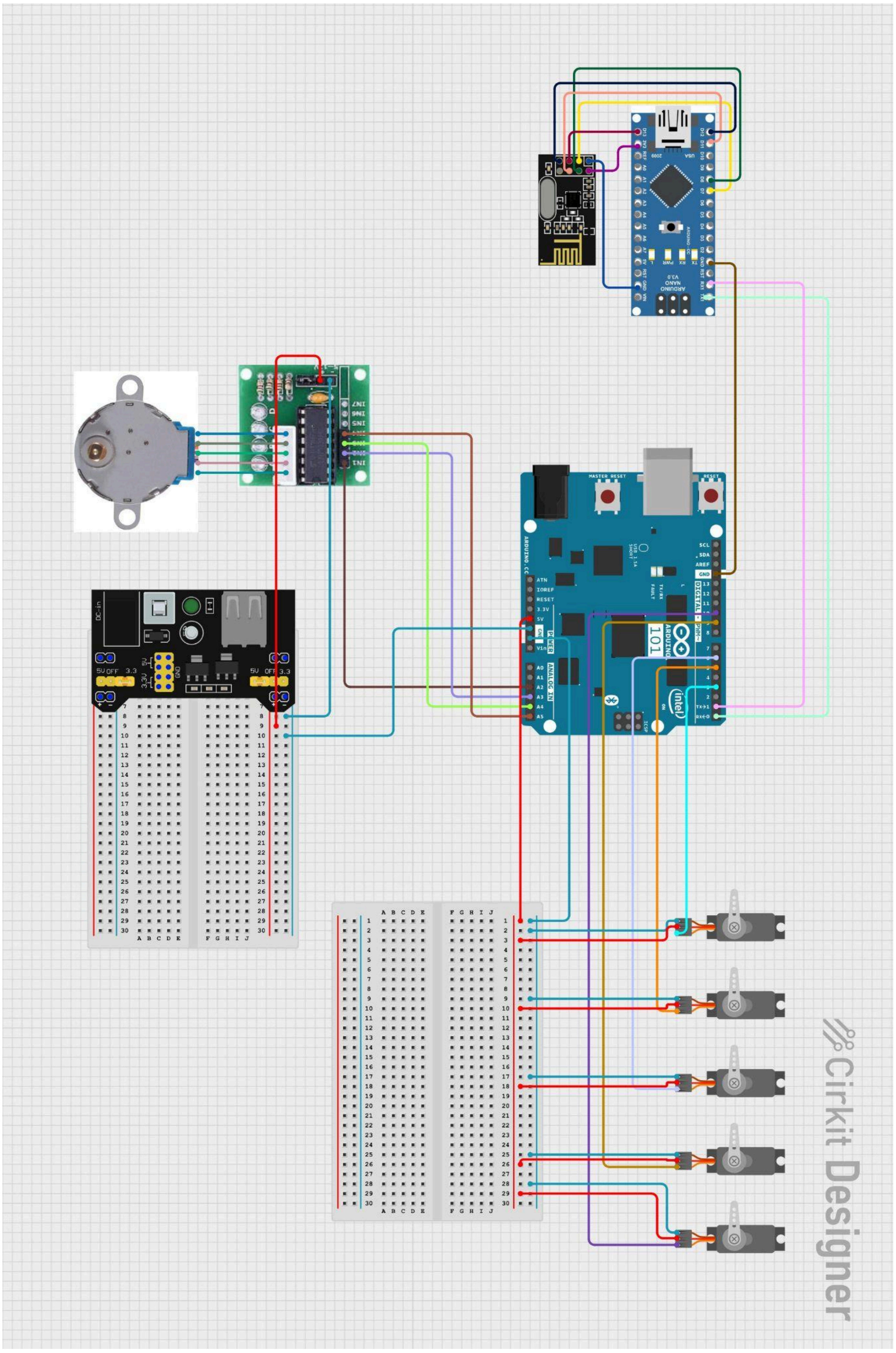
Packet Format

[UTF-8 text bytes][zero padding] $\rightarrow$ total = 32 bytes

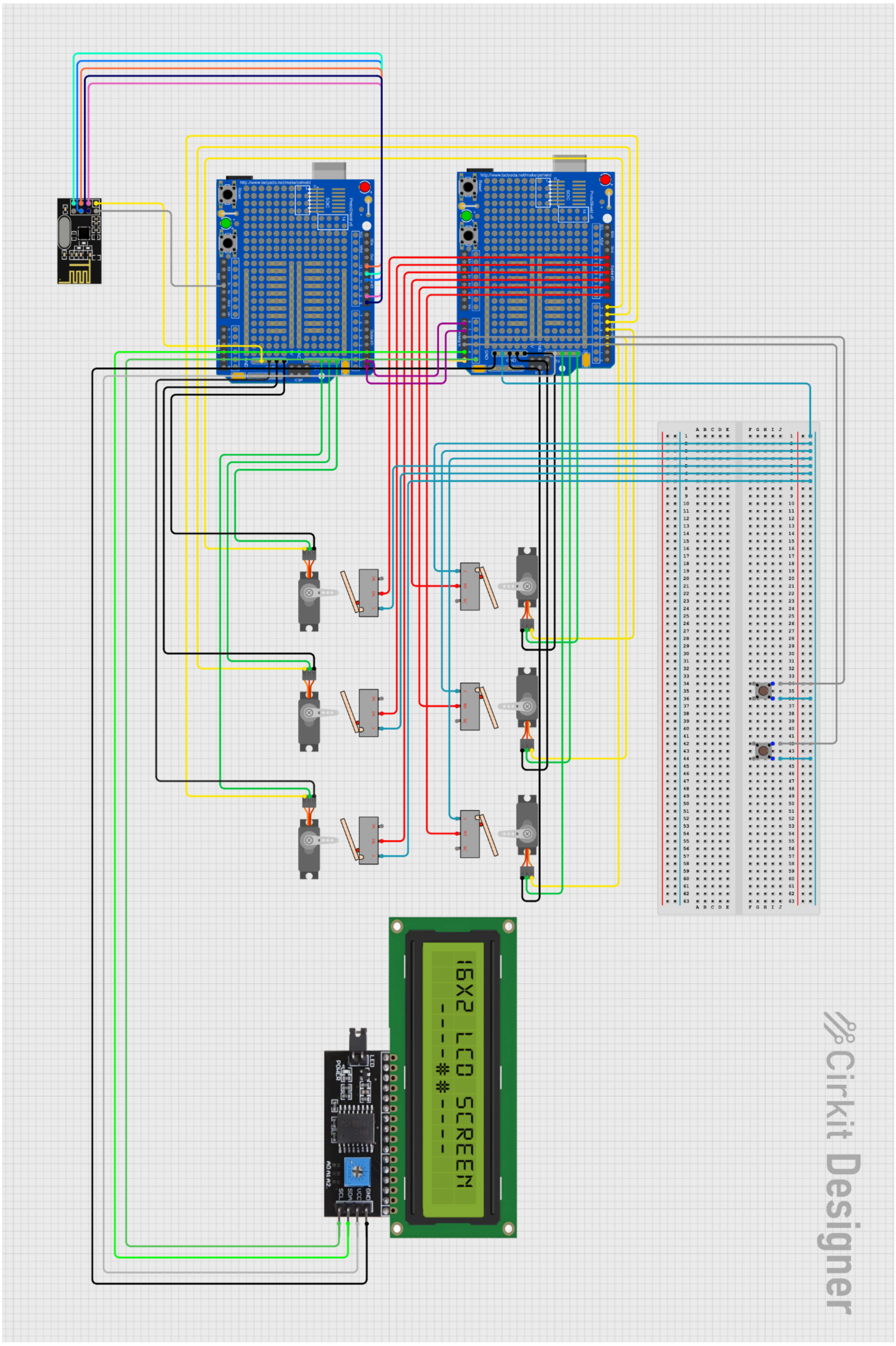
If a message exceeds 32 bytes, it is split into sequential chunks with a short delay between them. Receivers strip trailing zeros and use the text directly. Both Arduino modules receive **all** packets; each one decides locally whether to act on the text (hand checks for known words, Braille prints all letters).

Circuit Diagrams:

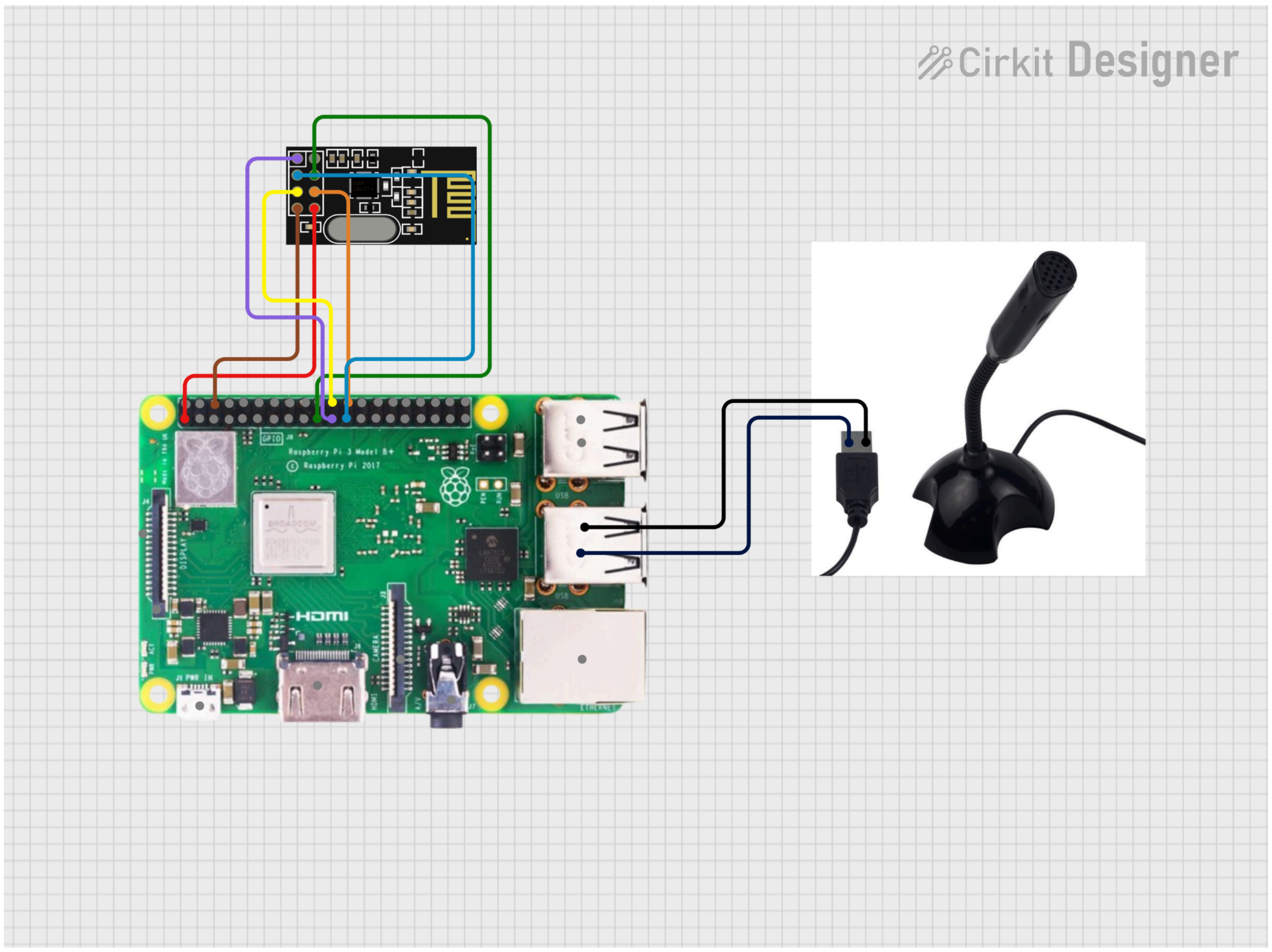
1. Hand



2. Braille



3. RPI and NRF module



Final setup

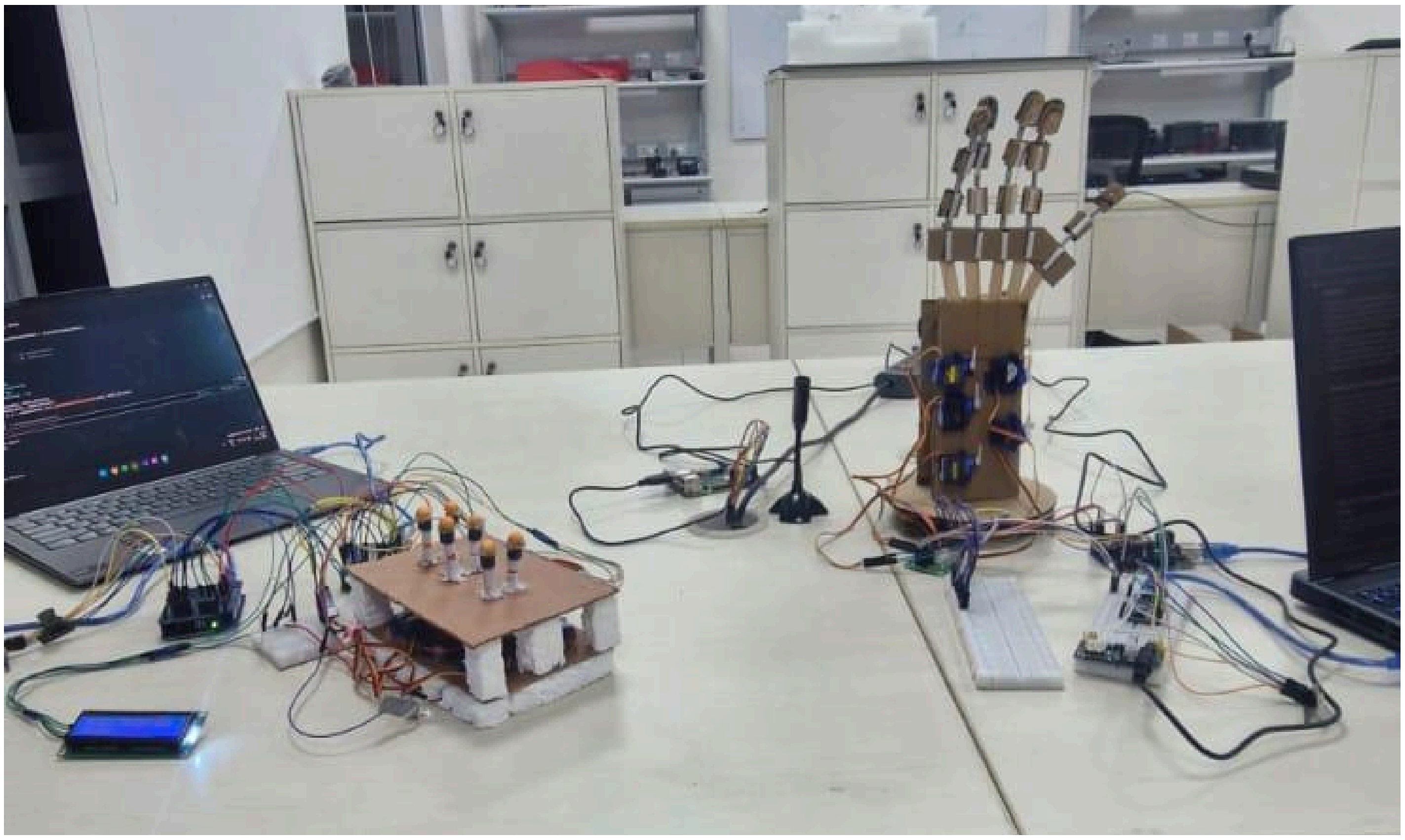


Fig. The above is how the final setup of the deliverables looks. On the left we have the braille system and on the right, we have the hand. The mic connected to the RPI along with the NRF module is the source of input for both hand and braille.

Link for Demo Video: https://drive.google.com/drive/folders/17wQdvqsMHzBrS9W2FIqWGcdnZbsrzA68?usp=drive_link