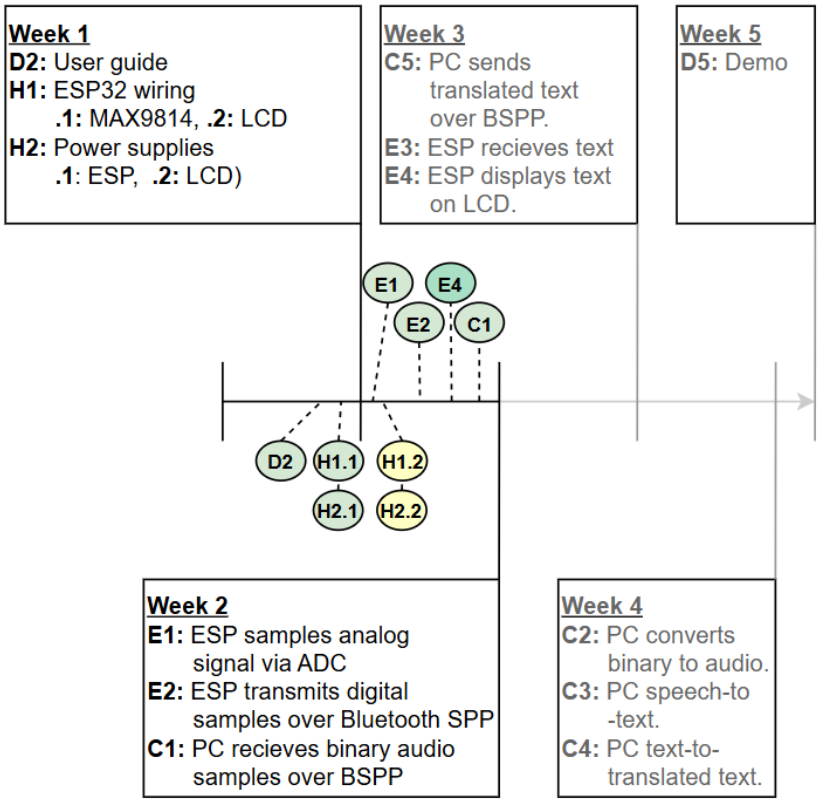
Overview

Original Objectives	Current Progress	Expected Date	Actual Date
User speaks into the mike	The MAX9814 is successfully wired.	**Week 1**	Week 1
ESP32 samples the analog singal via ADC	The sampling works, and my voice replay on the PC is comprehensible.	**Week 2**	Week 2
ESP32 sends digital samples over Bluetooth SPP to PC	The ESP32 has no issues transmitting data.	**Week 2**	Week 2
PC receives binary data	It currently works, but previously Windows did not cooperate for various reasons. It's not guaranteed that it will continue connecting reliably.	**Week 2**	Week 2
PC converts it to processable audio	(Partially done). Was able to reconstruct audio and the PC successfully relayed what I spoke into the mike (in decent time). However, the volume was very low, there was occasional spikes of white noise, and the format needed for upcoming processing still needs to be identified.	**Week 4**	Week 2
(PC) Speech-to-text	TBD	**Week 4**	
(PC) Text translation	TBD	**Week 4**	
(PC) Text transmission	TBD	**W. 3-4**	
ESP receives translation	TBD	**Week 3**	
ESP displays text on LCD screen	Successfully display text typed on the Serial Monitor on the LCD screen. However, it has a wrap-around issue, so a scrolling handler will need to be implemented (med priority, low effort).	**Week 3**	Week 2

Milestones



Lab

Week 1:

ESP32 & Mic Setup

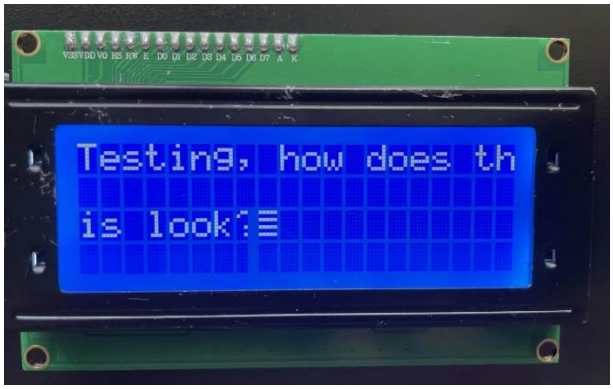
- Identified hardware (Different ESP board have different Bluetooth capabilities, so it matters)
    - DOIT ESP32 DEVKIT V1
    - MAX9814
    - I2C 20x4 LCD Display Module, with probably a PCF8574 I2C backpack
- Powered ESP32 through USB.
- MAX9814 - ESP32 wiring:
    - VCC – 3.3 V
    - GND – GND
    - OUT – GPIO35 (ADC1_CH7)

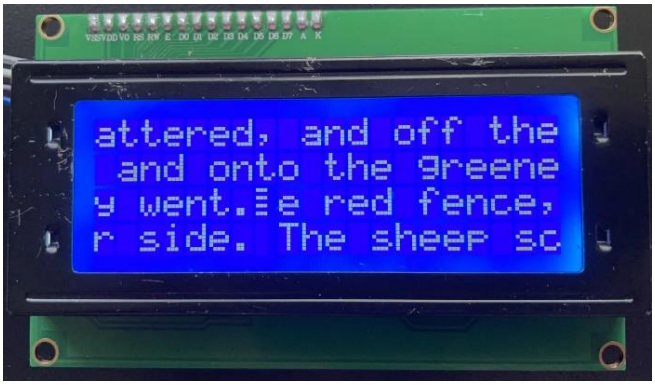
- Issue: ESP not detected by widows. Solves by installing CP210x USB to UART driver from Silicon labs.
- Issue: LCD drew too much current. Removed the LCD to handle power later.

Week 2:

Bluetooth Debugging and LCD setup

- Flashed ESP with server sketch and paired it over Bluetooth classic. Had trouble pairing (Windows showed the device as Not Connected or failed to assign virtual COM ports).
- Switched focus to LCD screen.
    - Wired LCD. Used battery pack for power.
      SDA-GPIO21, SCL-GPIO22, VCC – battery pack red, GND (lcd) – battery pack black – GND (esp)
    - Initially failed to detect I2C address. Fixed by replacing Wire.begin() with Wire.begin(21, 22) (GPIO pins for SDA/SCL). Eventually found LCD address 0x27.
    - Confirmed display works by having screen display anything typed into the serial monitor.

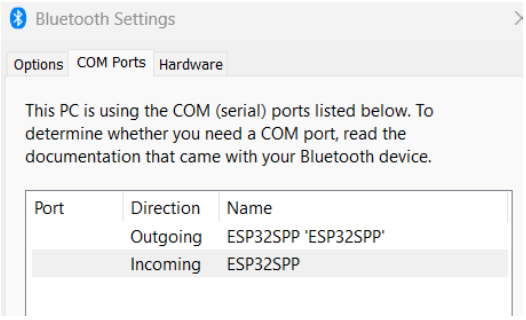




When there are large amounts of text, the LCD screen has an overflow issue. The LCD Driver object will need to implement scrolling logic.

- Bluetooth Issues: Windows kept showing it as “Not connected”, not assigning virtual incoming/outgoing ports, or failed to receive data. Solved by:
    - Restarting (removing device from Bluetooth list, restarting Bluetooth stack via services.msc, deleting old registry, renaming ESP device name back to old to match service name, restarting ESP with EN button), using PuTTy to establish ports.

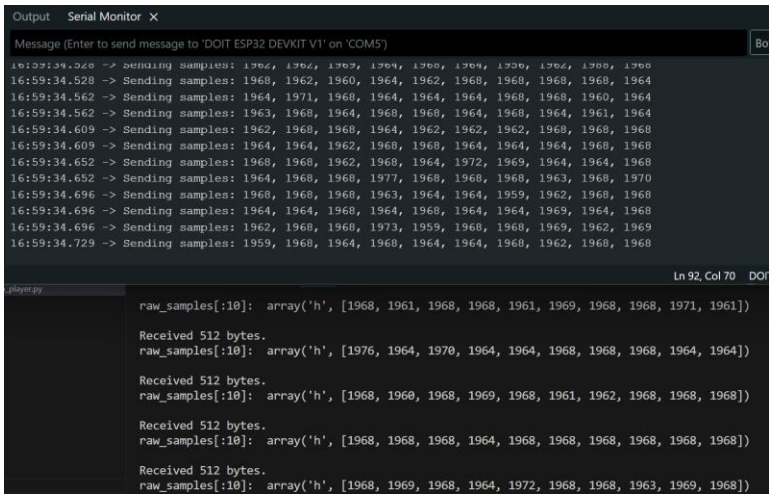
- Port fields were empty due to using old service name ‘ESP32SPP’ while new name was ESP_M. Changed name back since no amount of restarting/deleting on Windows got rid of it.



  - https://github.com/espressif/arduino-esp32/issues/5164#issuecomment-838509946 Uninstall esp32 libraries, reinstall version 1.0.4.
  - Set esp as server with false flag in ‘SerialBT.begin(NAME, false)’.

- Issue: Audio was noisy and under-sampled. Switched to using I2S with DMA for faster ADC reads.
- Issue: Output stalls after connection. Fixed by slowing does ESP32 serial output, configuring ADC before the I2S setup, setting pin 35 as INPUT, and removing the blocking call “adc1_get_raw(...)” (which was intended for debugging).

- Audio replay finally worked. The replay got faster over time, but there were occasional spikes of white noise and audio volume was low.



Plans for Next Week

- **C5:** Implement Bluetooth-based transmission of ~~translated~~ text from PC->ESP (currently using USB).
- **E3:** Test that ESP is able to receive incoming text over Bluetooth.
- **E4:** Integrate the LCD display logic to show the received text.
    - Implement scrolling with small delay to handle text overflow.
- Need to check that the Bluetooth setup is reliable across reconnections.
- :Stretch goal: Move from individual test sketches and begin putting together the pipeline.

Definition of Done

The project is good enough when these principles use cases are successfully implemented:
1. The ESP32 captures audio via the microphone.
2. Audio samples are transmitted from the ESP32 to the PC over Bluetooth SPP.
3. The PC performs speech-to-text transcription on the received audio.
5. The transcribed text is sent back to the ESP32 and displayed on the LCD screen.

The project is considered fully Done when the PC performs text translation before sending the final output to the ESP32 for display.

Scope Modifications

The scope does not need modifications yet with milestones still mostly on track. However, if Bluetooth communication remains unstable, it might be necessary to drop PC-side processing on Week 4.

One issue is that documentation work has not been assigned time, so D5 (demonstration) is at risk of being incomplete. Given it's a higher priority than PC-side-processing, the D5 milestone is shifted to Week 4, and C4 (PC-side translation) is pushed to Week 5.