BLM4021 Gömülü Sistemler
Laboratuvar Projesi Final Raporu

Grup No: 4

Proje Kategorisi: Uyg 2

Kişilerin Çalışma Yüzdesi:

Grup Sorumlusu:	**Yavuz Selim Çağan**	**25**	**OLED Ekran Kontrolü, TCP Server ve Process Çalıştırıcı**
	Umut Can Sevdi	25	**OLED Ekran Kontrolü, TCP Server ve Process Çalıştırıcı**
	Semih Yazıcı	25	**Ses Kaydı, ve Ses İşleme ve TCP Client**
	Oğuzhan Ercan	25	**Ses Kaydı, ve Ses İşleme ve TCP Client**

Yukarıda sadece kırmızı yerleri doldurunuz / güncelleyiniz.

İçerik

Bu bölüme dokunmayınız. Sayfalar hiç kaymayacak ve ilgili bölüm hep aynı sayfada başlayacaktır. Kullanmadığınız sayfaları boş geçiniz.

$$\sum 18 SAYFA$$

I. Giriş ve Proje Tanıtımı

Projemizde hedefimiz ses kayıtları göndererek Linux yüklü bir cihazı kontrol edebileceğimiz bir sistem oluşturmak. Projemizde sonuçları OLED display'de görüntülemek istedik.

4 kişi olarak çalıştığımız projemizde görev dağılımını ilgili olduğumuz alanlara göre yaptık.

Yavuz Selim Çağan ve Umutcan Sevdi C/C++ ile daha önce çalışmış oldukları için TCP sunucusu ve OLED ekranın yazılımlarını geliştirdiler. Bu program birbirlerini durdurmamaları için ayrı iki processte çalışmaktadır ve kendi aralarında pipe ile iletişim kurmaktadır. Programlardan biri sürekli olarak 8081. porta gelen TCP mesajlarını dinlemektedir, gelen mesajları çalıştırmak üzere pipe üzerinden göndermektedir. Diğer ise gelen programı çalıştırmakta ve sonucunu OLED ekrana basmakla görevlidir. Displayden sorumlu program her çalıştıracağı program için ayrı bir kısa ömürlü process açmakta ve sonucunu pipe ile okumaktadır.

Semih Yazıcı ve Oğuzhan Ercan ise ses işleme kısmını geliştirdi. Bu kısım ise Python dilinde yazıldı. Bu proje ses işlemenin yanında aynı zamanda bir TCP client olarak davranmaktadır. Mesajları anlamlandırıp uygun komutlarla değiştirdikten sonra TCP sunucusuna göndermektedir.

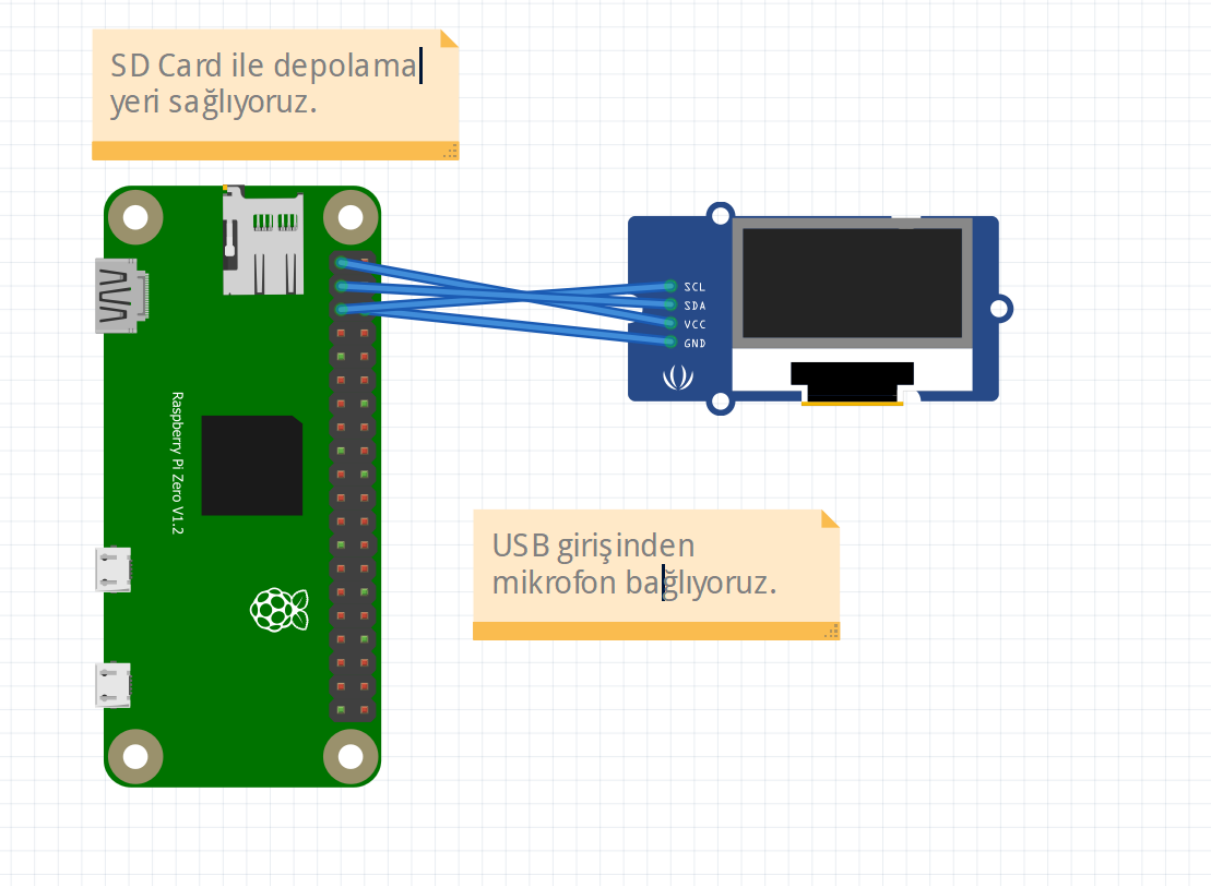
Python programı 3 farklı komponentten oluşmaktadır. Birinci komponent ses dinleme modülüdür. sounddevice modülü aracılığı ile belirli bir süre boyunca ses dinlenir ve wav formatında kaydedilir.

İkinci komponent sesi metine çevirmektir. wav formatında kaydedilen ses belleğe alınır ve speech_recognition modülü aracılığı ile Yahoo Speech2text api'ına gönderilir. Yahoo Speech2text api'ı sesi metne çevirir ve metni programımıza gönderir. speech_recognition modülünden çağırdığımız fonksiyon metni döndürür.

Üçüncü komponent ise eşleştirme işlemidir. Speech2text'ten gelen ham metin bu komponentte işlenir. Bu işlemin temel amacı ham metini komut formatında çevirmektir. Örneğin change directory olarak söylenen komut dictionaryde komut karşılığı olan cd'ye çevirilir. Bu işlem sonucunda oluşturulan metin C tarafındaki tcp sunucusuna iletilir.

-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-

II. *Fritsing* ile Ön Tasarım



Şekil 1: Fritsing Öntasarımı

Yukarıda gösterildiği gibi OLED ekran ile Raspberry 4 pin üzerinden birbirlerine bağlanmışlardır. Raspberry Pi'n 1. 3. 5. ve 6. pinleri kullanılmıştır. 1.pininden 3.3V'luk bir gerilim elde edilmiştir. 3.pininden iletişimin gerçekleşmesi için data akışı sağlanmıştır. 5.pininden seri haberleşmenin düzgün bir şekilde gerçekleşmesi için ortak bir clock kullanılması sağlanmıştır. Son olarak OLED'in ground pini Raspberry Pi'in 6.pini olan ground pinine bağlanmıştır.

Raspberry Pi'a yukarıda gösterilmeyen lakin sesin elde edilmesi için kullanılan bir mikrofon bağlanmıştır. Bu mikrofon Rasbperry Pi'a Micro-USB aracılığı ile bağlanmıştır.

Raspberry Pi’a işletim sistemi olarak Linux tabanlı Raspbian kurulmuştur. Bunun için ise 32GB depolama alanın sahip bir SD Card takılmıştır.

-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-

III. Kurulan Devre Detayları

OLED 0.96 inç 128x64 Piksel Ekran Modülü - Mavi Sarı :

OLED ekran modülü 0.96 inç diyagonal boyutunda olmakla birlikte 128x64 çözünürlüğe sahiptir. Görüş açısı: 160 ° 'den büyüktür ve destek voltajı: 3.3V-5V DC arasındadır. Güç tüketimi, normal çalışma sırasında 0.04W en yüksek performansta 0.08W yakmaktadır.

Gömülü sürücüsü IC: SSD1306, iletişim ise I2C/IIC şeklinde sağlar. Cihaz, sadece iki i/o bağlantı noktasına ihtiyaç duyar. Bu OLED ekran Raspberry Pi tarafından rahatça kullanılabilmektedir. Bir bağlantı CLK için diğer bağlantı ise Data için kullanılmaktadır.

Arka ışık gerekmez, ekran ünitesi kendini aydınlatabilir. Süper yüksek kontrast, parlak ve net noktalar, hatta küçük yazı tipleri oldukça okunabilir durumdadır.

OLED denetleyicinin içinde gömülü yazı tipi yok, kullanıcı yazı tiplerini yazı tipi oluşturma yazılımı aracılığıyla oluşturabilir.

Mikrofon:

Gelişmiş bir kulak üstü kulaklık kullanılmıştır. Micro-USB girişi sayesinde Raspberry Pi’a bağlanmıştır. Ses paketleri python üzerinden elde edilip işlenmiştir.

Raspberry Pi Zero:

Boyutu çok küçüktür. Bu yüzden tam boy USB veya HDMI bağlantıları bulunmamaktadır. Onun yerine Micro-USB ve Micro-HDMI girişleri bulunmaktadır.

İşlemcisi 1GHz frekansa sahiptir. ARM11 çekirdek bulunmaktadır. Raspberry Pi 1 modeline göre %40 daha hızlıdır.

512 LPDDR2 RAM'e sahiptir.

Mikro-SD kart yuvası bulunmaktadır.

1080p60fps Micro-HDMI video çıkışı vardır.

Bugüne kadarki en küçük boyutlu Raspberry Pi'dır. (65mm x 30mm x 5mm).

40-pin GPIO yuvası bulunmaktadır.

Bağlantılar:

OLED VE Raspberry Pi Zero bağlantıları Raspberry Pi'ın 1. , 3. , 5. ve 6. pinleri kullanılarak gerçekleştirilmiştir. Bağlantı yollarının açıklaması 2.bölümde anlatılmıştır. 1.pin sayesinde OLED'e 3.3V'luk bir gerilim sağlanmıştır. 3.pin sayesinde I2C ile gerçekleşecek olan veri akışı için bir yol oluşturulmuştur. Bu sistemde OLED slave Raspberry Pi Zero ise master durumundadır. 5.pin ile CLOCK sağlanarak iletişimin düzenli gerçekleşmesi sağlanmış ve 6.pin OLED için GROUND olarak ayarlanmıştır.

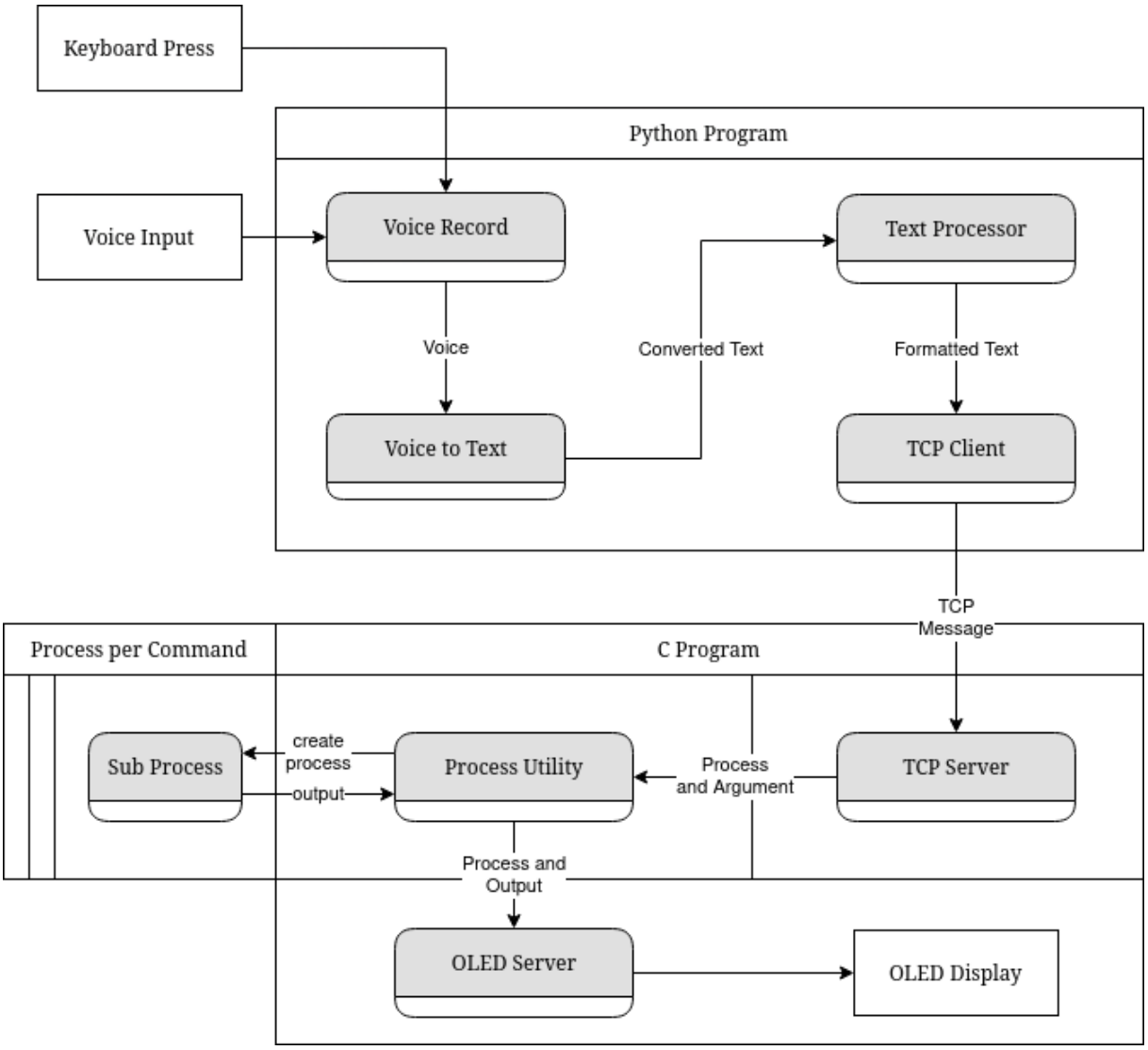
Mikrofon Micro-Usb aracılığı ile bağlanmıştır.

32GB SD card takılmıştır.

-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-

IV. Yazılım Tasarımı

Projemizin yazılım kısmı iki aşamadan oluşmaktadır. Bunlardan ilki sesi kaydeden ve işleyen kısımdır. Diğeri ise gelen komutu çalıştıran ve sonucu OLED ekrana bastıran kısımdır. Genel akış şeması Şekil 2'de belirtilmiştir.



Şekil 2: Yazılım Sistem Dizaynı

Şekil 2'de her kutu ayrı bir processi temsil etmektedir. Her oval ayrı bir fonksiyonu Boş kutular ise I/O'yu temsil etmektedir. Aradaki oklar iletilen mesajı belirtir.

1. Ses Kaydı Programı:

Programımız Python dilinde yazdığımız ses işleme programımızı çalıştırabilmek için aşağıdaki kütüphaneler gerekmektedir:

- numpy
- SciPy
- sounddevice
- SpeechRecognition
- PyAudio

Ayrıca bu kütüphanelerin Linux'ta çalışabilmesi için extra paketlere ihtiyaç duyulmaktadır. Bunları ise

- *sudo apt install portaudio19-dev python3-pyaudio*

komutu ile indirebiliriz.

Ardından programımız aynı ağ üzerinde kendi hostname'ine denk düşen IP adresini elde edecektir. Bu ip adresinin 8081. portuna socket API ile bağlanacaktır. Bunun sebebi ise C programında yazdığımız TCP sunucusunun 8081 portunda çalışıyor olmasıdır. Bu bilgileri elde ettikten sonra C programının bulunduğu sockete bağlanır. Porta bağlanmasından sonra ses işlemede kullanacağımız Yahoo API tanımlanır.

Bundan sonra ise kodumuz sonsuz döngüye girer. Döngü sırasında klavyeden bir input verildiğinde 5 saniye boyunca sesi kaydeder. Ses kaydı

*data = sd.rec(samplerate * duration, samplerate=samplerate, channels=1, blocking=True)*

ile gerçekleştirilir. Ve elde edilen ses diske yazılmadan önce uygun bir aralığa

*data= (data * abs_max + abs_max).clip(0,abs_max).astype(np.int16)*

şeklinde getirilir. Ardından sesin diske yazılması için bir süre beklenir. Bu işlemin ardından sesi dosyadan okuyup Google TTS API'a iletir.

Gelen cevabın durumuna göre mesaj *text_processor(text:str)* fonksiyonundan geçirilir. Bu fonksiyon gelen mesajı çalıştırılabilecek bir komut formatına getirir. Eğer kelimeler önceden tanımlanmış komut dictionary'sinde bulunmaktaysa onlarla değiştirilir.

text_processor fonksiyonundan sonra elde edilen son mesaj null değilse karşı C programına TCP üzerinden iletilir. Herhangi bir anda hata alınması durumunda try catch blokları ile yakalanır ve programa devam edilir. Örneğin eğer gelen mesaj anlamandırılamazsa "Sorry, I didn't catch that." mesajı ekrana basılır.

2. **OLED Display ve Process Çalıştırıcı Program:**

C dilinde yazdığımız programımızın çalıştırılabilmesi için bazı kütüphanelerin kurulu olması gerekmektedir. Bunlar:

- SSD1306_OLED_RPI
- bcm2835

Programız mesajları almaya devam ederken OLED ekranı işlemeye devam edebilmesi için multi process olarak yazılmıştır.

İlk processimiz OLED ekranın kontrolü ve programların çalıştırılmasından sorumlu iken ikinci processimiz TCP server olarak görev yapmaktadır.

Programımız iki farklı processte çalışacağı için iki program arasındaki iletişimde *pipe* yapısı kullanılmaktadır. TCP server pipe'ın yazma ucunu display programımız ise okuma ucunu almaktadır.

2.1 Display Process:

Bu processimizin öncelikle çeşitli ayarlarının yapılması gerekmektedir. OLED displayin ayarlanması için *int disp_setup(pid_t p)* fonksiyonumuz çalıştırılmaktadır. Bu fonksiyon OLED ekranın programa tanıtılmasını sağlamaktadır ve verilen *disp_width* ve *disp_height* değerlerine göre bir SSD1306 class'ı oluşturmaktadır. Bu class display ile ilgili fonksiyonları içerir. Ardından ekrana tüm renkleri basıp bir süre bekledikten sonra ekranı temizler.

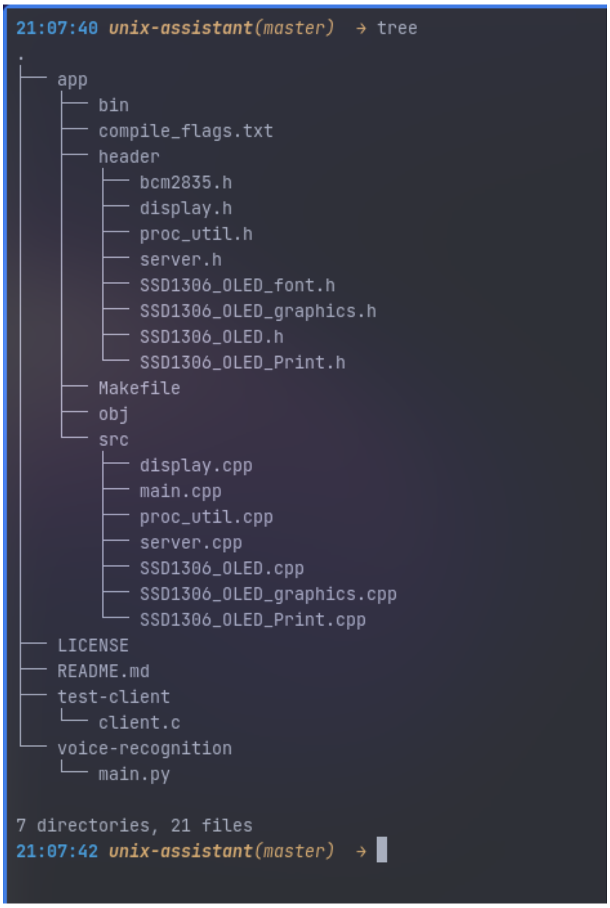
Bunun ardından void *disp_loop(int read_pipe)* fonksiyonu çalışır. Bu fonksiyon ekrana başlangıç için bir text yazdırır ve pipe'ın okuma ucunu dinlemeye başlar. "exit" dışında bir mesaj geldiği sürece şu işlemler uygulanır.

- Eğer mesaj "clear" ise OLED ekran temizlenir.
- Eğer değilse display cursor'u y_index * 8 % 64 formülü ile olması gereken yere yerleştirilir. Ardından gelen komut proc_util'de tanımladığımız *char *proc_exec(char *const program, int len)* fonksiyonunu çağırır.
- "proc_exec" öncelikle yine aynı headerda tanımlı *char **proc_split(const char *const string, int len, int *argc)* fonksiyonunu çağırır. Bu fonksiyon gelen stringi program ve argümanları şeklinde parse eder.
- Parse işleminin tamamlanmasından sonra *proc_exec* yeni bir process yaratır. Process'e bir pipe ile bağlanır. Bu process'in argüman ve program verilerini girerek *exec* sistem çağrısında

bulunur. Pipe’ın write ucuna stdout ve stderr kanallarını yönlendirir. Bu sayede oluşturulan processin outputu bir stringe atanabilir.

- Process tamamlandığında pipe’dan çıktılar okunur ve process sonlanır.
- Program adını ve sonucu elde edilen disp_loop bunları OLED buffer’ına yazar ve ekranı yeniler. Eğer ekran bu sonuçta overflow olduysa ekran temizlenir.
- Eğer mesaj “exit” ise, shell’deki exit yerine OLED ekran ve TCP server processleri sonlandırılır. OLED’in gücü kesilir ve kullanılan network port’ları kapatılır.

Kodun geliştirilme sürecini kolaylaştırmak için modüler olarak geliştirdik.



Şekil 3: Projenin Dosya Yapısı

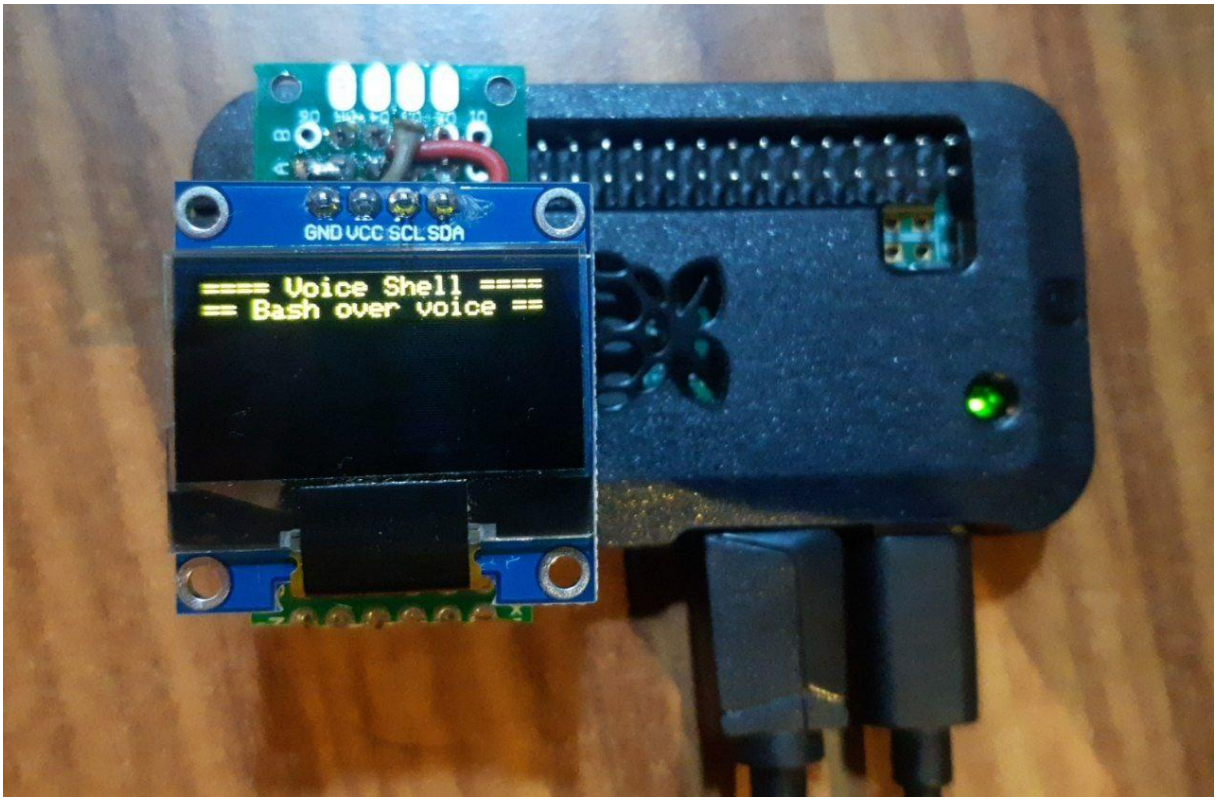
-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-

V. Sonuçlar, Demo Detayları ve Sunum Linki

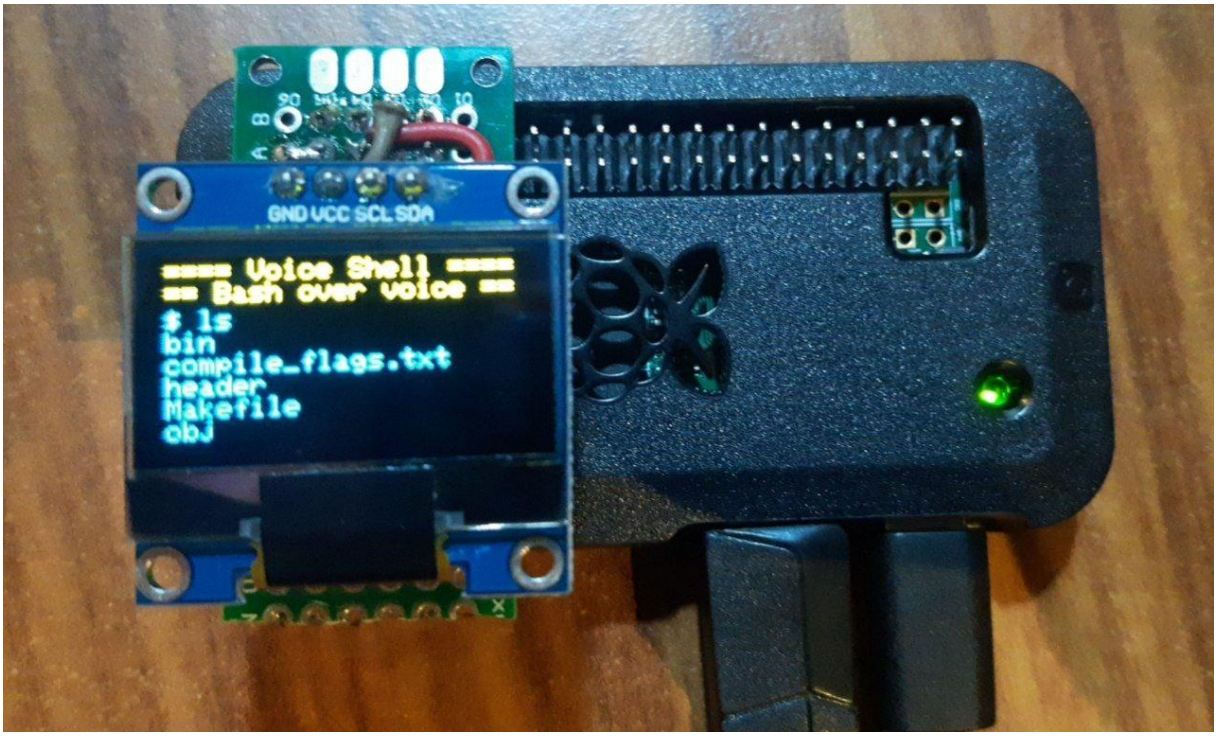
Proje sonucunda ses ile kontrol edilebilen ve sonuçları OLED ekrana yansıtan sistemimiz başarı ile tamamlandı. Sesin anlaşılır olmaması veya gürültüden dolayı belirli durumlarda komutlar istenen şekilde anlamlandırılamayabilir. Bazı komutlar beklenenden daha uzun output verdiklerinde yazılar üst üste gelebilmektedir. Bu durum dışında sesler başarılı bir şekilde çevrilmekte iletilmekte ve program olarak çalıştırılabilmektedir.

Proje'nin sunumuna https://www.youtube.com/watch?v=t-gM7oC8kfU adresinden ulaşabilirsiniz.

Aşağıdaki görsellerde ise projenin çalışma anında çekilen görselleri görebilirsiniz.



Şekil 4: OLED Display Açılışı



Şekil 5: “ls” Komutu Çalıştırılıyor

Şekil 6: “uname” Komutu “--all” Argümanı İle Çalıştırılıyor.

-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-

VI. Referanslar

[1] Gavin Lyons "SSD1306_OLED_RPI" 2021, https://github.com/gavinlyonsrepo/SSD1306_OLED_RPI.

[2] Broadcom "Broadcom BCM 2835 Chip Library" 2012, https://www.airspayce.com/mikem/bcm2835/.

[3] Hubert Pham "PyAudio" 2006, https://pypi.org/project/PyAudio/.

[4] Travis E. Oliphant et al. "Numpy" 2006, https://pypi.org/project/numpy/.

[5] Guido van Rossum "Socket" 1990, https://docs.python.org/3/library/socket.html.

[6] Anthony Zhang (Uberi) "Speech Recognition" 2014, https://pypi.org/project/SpeechRecognition/.

[7] Raspberry Pi "Raspberry Pi Zero W" 2022 https://www.raspberrypi.com/products/raspberry-pi-zero-w/

-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-

-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-