PianoHand

- **Author: Bigan Radu-Cristin**
- **Group: 335CA**

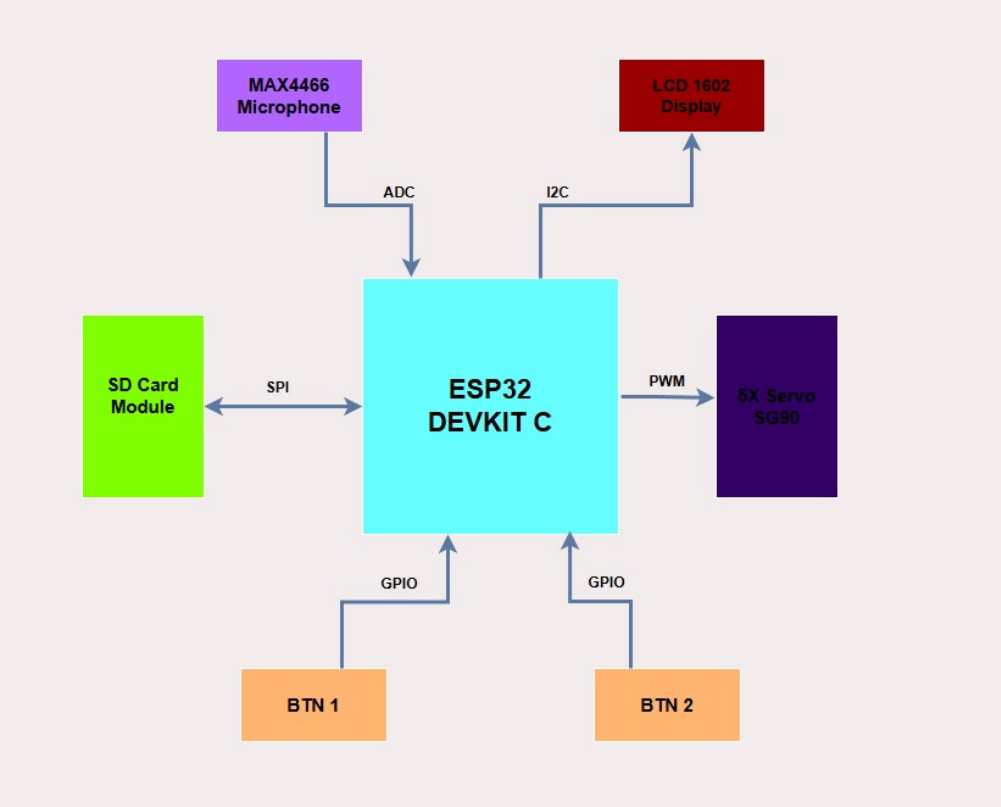
Introduction

This project consists of building a robotic hand with 5 fingers, each actuated by a servo motor, placed over 5 piano keys (DO, RE, MI, FA, SOL). The system can detect musical notes from different sources and reproduce them physically on the keys.

- **What it does:** The hand listens to or reads music and plays it back on an electronic keyboard by pressing the keys with servo-driven fingers. It supports three operating modes: playing a hardcoded melody from memory, listening to a short melody through a microphone and reproducing it, or reading a .wav file from an SD card and playing the detected notes.
- **Purpose:** The goal is to explore the integration of audio signal processing (FFT-based note detection) with real-time motor control on an embedded platform, while also covering a wide range of communication protocols studied in the PM course.
- **Starting idea:** The idea came from wanting to build something that bridges digital signal processing and physical actuation — a robot that can actually “hear” music and play it back.
- **Why it is useful:** It demonstrates a practical application of FFT on a microcontroller, combining audio analysis with mechanical output.

General Description

System Block Diagram:



The system is centered around an ESP32 DevKit C microcontroller that manages three main tasks: user interaction (buttons + LCD display), audio input/analysis (microphone ADC or SD card SPI), and mechanical output (5 servo motors via PWM).

The user selects between the three operating modes using two physical buttons, and the current mode and status are shown on an LCD 1602 display. Depending on the selected mode:

1. **Mode 1 — Hardcoded Melody:** The firmware contains a pre-stored sequence of notes with timing information. The ESP32 iterates through the array and drives the corresponding servo for each note.
2. **Mode 2 — Microphone Input:** The MAX4466 electret microphone module captures ambient audio. The analog signal is sampled via the ESP32's internal ADC, then processed through a 2048-point FFT pipeline to detect the dominant frequency. The detected frequency is matched to the closest musical note using a logarithmic (semitone-based) chromatic scale comparison. A histogram of all detected notes is built automatically, and the top 5 most frequent notes are mapped to the 5 fingers.
3. **Mode 3 — WAV File from SD Card:** A .wav file (8-bit mono PCM) stored on a Micro SD card is read over SPI. The same FFT pipeline from Mode 2 is applied to the audio data to extract notes, which are then played on the servos.

Component	Protocol	Direction	Role
MAX4466 Microphone	ADC (analog)	Microphone → ESP32	Audio capture for Mode 2
SD Card Module HW-125	SPI (MISO, MOSI, SCK, CS)	Bidirectional	WAV file storage for Mode 3
LCD 1602 Display	I2C (SDA, SCL)	ESP32 → Display	Shows current mode, active note
5× Servo SG90	PWM (LEDC channels)	ESP32 → Servos	Actuate fingers on piano keys
BTN1	GPIO (digital input)	Button → ESP32	Start/Stop playback

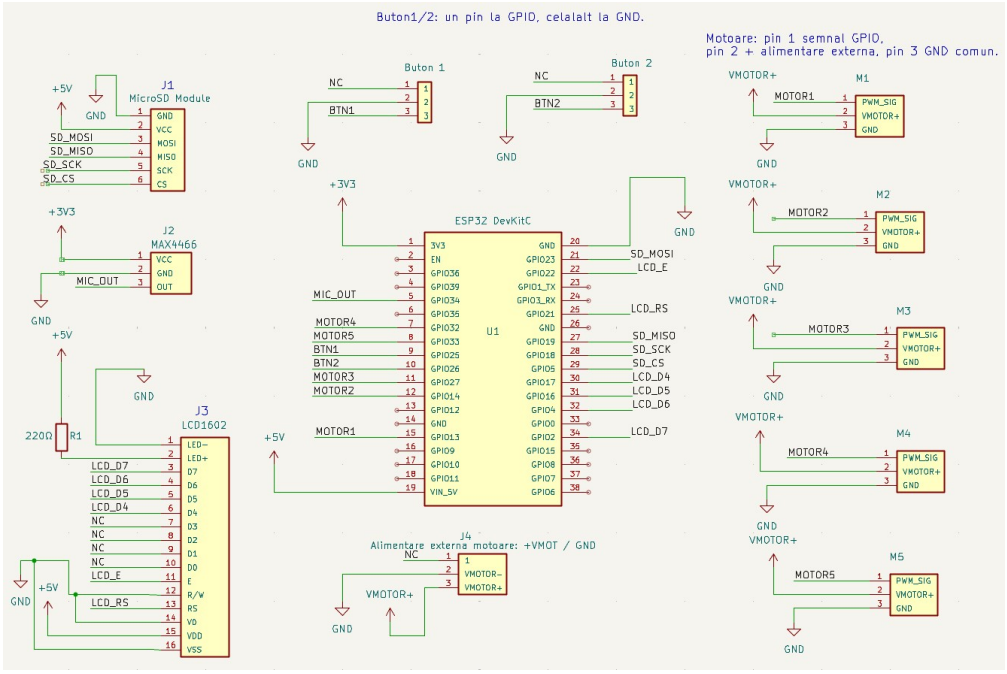
BTN2	GPIO (digital input)	Button → ESP32	Cycle between modes

Hardware Design

Bill of Materials:

Component	Quantity	Role
ESP32 DevKit C (WROOM-32)	1	Main microcontroller
Servo Motor SG90	5	Finger actuation (one per piano key)
MAX4466 Electret Microphone Module	1	Audio capture (Mode 2)
SD Card Module HW-125	1	WAV file storage (Mode 3)
LCD 1602 Display (with I2C backpack)	1	User interface display
Tact Switch Button 6×6mm	2	Mode selection and start/stop
220Ω Resistor	1	Current limiting for the LCD backlight
Breadboard 830 points	1	Solderless assembly
External 5V/2A Power Supply	1	Servo power

Electrical Diagram:



Software Design

Descrierea codului aplicaţiei (firmware):

- mediu de dezvoltare (if any) (e.g. AVR Studio, CodeVisionAVR)
- librării şi surse 3rd-party (e.g. Procyon AVRlib)
- algoritmi şi structuri pe care plănuiţi să le implementaţi
- (etapa 3) surse şi funcţii implementate

Rezultate Obţinute

Care au fost rezultatele obţinute în urma realizării proiectului vostru.

Concluzii

Download

O arhivă (sau mai multe dacă este cazul) cu fişierele obţinute în urma realizării proiectului: surse, scheme, etc. Un fişier README, un ChangeLog, un script de compilare şi copiere automată pe uC crează întotdeauna o impresie bună .

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** şi **Resurse Hardware**.

Export to PDF