Polyphonic Keyboard Synthesizer

Embedded Rust on STM32

Project Documentation & Implementation Plan

Platform:	STM32 Nucleo
Language:	Rust (Embedded)
Features:	Polyphony, MIDI-style Logic, Real-Time Audio

December 17, 2025

Contents

1 General Description

This project aims to develop a fully functional digital musical instrument capable of rendering polyphony (chords) and responding in real-time to user interactions (keys, sustain pedal, pitch bend).

The core of the system is an **STM32 Nucleo** microcontroller, programmed in the **Rust** language (using a Real-Time framework) to guarantee minimal audio latency and high reliability.

1.1 Functionality

- **Polyphonic Input:** Reading approximately 61 keys (organized in an $R \times C$ matrix) with anti-ghosting protection (diode per switch).
- **Audio Synthesis:** Digital generation of sound waves (sine/square) on the microcontroller, utilizing the DAC or PWM peripheral.
- **Expressive Control:** Implementation of the *Sustain* function (pedal) and *Pitch Bend* control (joystick).
- **Audio Output:** Amplifying the weak digital signal from the MCU to drive an external speaker.

1.2 Project Motivation

- **Technical Challenge:** The project combines electronics (key matrix, audio filtering) with real-time programming, serving as an excellent demonstration of Rust capabilities in the `no_std` (embedded systems) environment.
- **Experience Quality:** The choice of high-quality mechanical switches (Outemu Linear, 65g) over simple buttons offers a superior feel suitable for a musical instrument.
- **Practical Application:** Creating a functional digital instrument capable of playing complete songs.

2 System Architecture

2.1 Hardware Architecture

The system is divided into three main interconnected blocks:

1. **Input Block (Keyboard):** The $R \times C$ matrix of switches and diodes.
2. **Control and Logic Block (STM32):** Processing input and generating the digital audio signal.
3. **Output Block (Audio):** Filtering, amplification, and playback of the sound via the speaker.

2.2 Software Architecture

The Rust code uses a Task-based concurrency model (e.g., RTIC or Embassy) to manage audio input and output.

- **Audio Task (High Priority):** Runs at a fixed frequency (approx 44 kHz), managing the Timer and DMA to feed the DAC with sound wave samples.

- **I/O Task (Low Priority):** Periodically scans the key matrix and reads the ADC pins (joystick, pedal).
- **Logic Task:** Calculates the frequency (note) and amplitude (volume) based on I/O data and sends the parameters to the Audio Task.

3 Hardware Design (Circuit Design)

3.1 Critical Components

Component	Specifications	Role in the Circuit
Microcontroller	STM32 Nucleo (e.g., F401RE)	Processing, Timers, ADC, DAC/PWM.
Switches	Outemu Linear (3-pin), 65g	Ensures quality tactile input.
Polyphony Diode	1N4148	Prevents ”ghosting” in the $R \times C$ matrix.
Amplifier	PAM8403 Module (2×3 W)	Audio amplification from 3.3V to speaker level.
Matrix Wiring	Set 5 Spools 24 AWG	Stable, soldered wiring for the approx 70 internal connections.

Table 1: Critical Hardware Components

3.2 Logical Interconnection Diagram

The interconnection diagram illustrates how the hardware blocks connect to the STM32 peripherals.

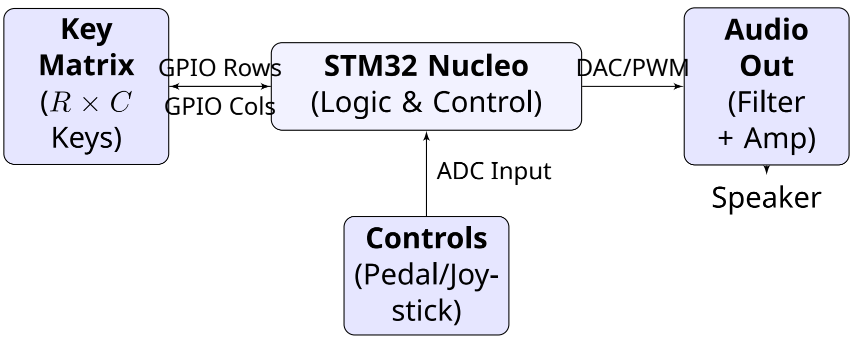

Figure 1: Logical Interconnection Diagram

3.3 Detailed Electronic Schematic (Simplified)

The electronic schematic provides a circuit-level view of the key components.

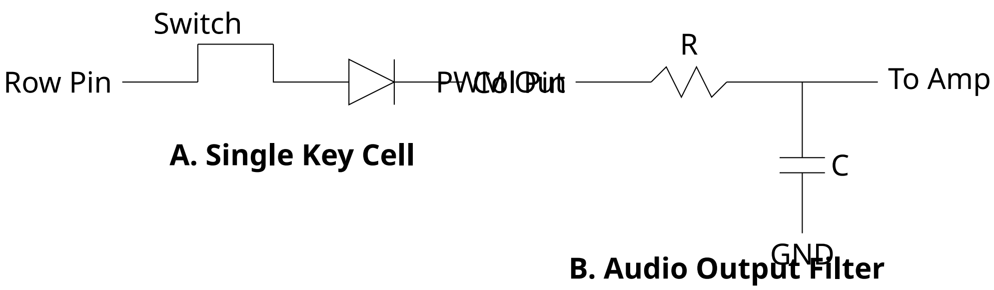

Figure 2: Simplified Schematic: (A) Matrix Switch with Diode, (B) RC Low-Pass Filter

3.4 Prototyping and Wiring

- **Matrix Construction:** Will be built by manually soldering the 24 AWG cable to the output pin of each switch.
- **Termination:** The approx 20 wires from the matrix will be soldered onto the 20-pin IDC Connector.

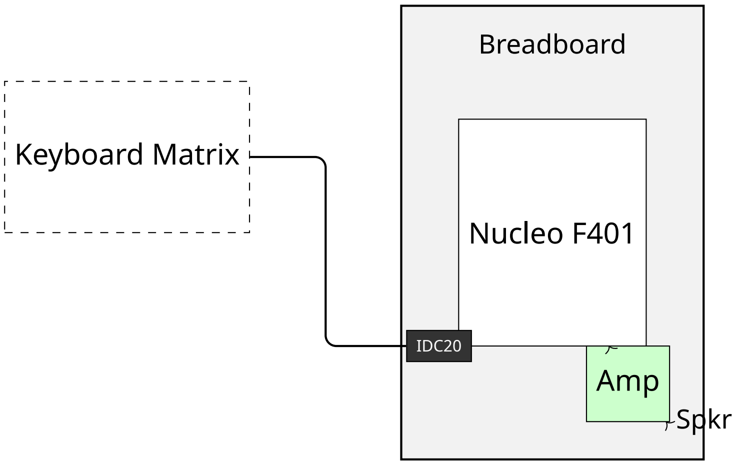

Figure 3: Physical Wiring Concept

4 Weekly Log (Project Status)

This log tracks the project's progress, from component acquisition to the implementation of key functionalities.

Week 1: Planning and Hardware Acquisition

Date	Status	Observations
Day 1-2	Finalized BOM	Final confirmation of components (Outemu, PAM8403, 1N4148).
Day 3-5	Workspace Setup	Installation of Rust toolchain (`cargo-embed`, `probe-rs`).
Day 6-7	Hardware Design	Creation of logic schematic and interconnection diagram.
Status:	**90% Complete**	Component acquisition is the priority.

Week 2: Basic Tests and Soldering

Date	Status	Observations
Day 8-10	Matrix Soldering (Ph.1)	Soldering first 20 diodes. Wiring 4×4 test matrix.
Day 11-12	GPIO/Input Test	Rust code to scan 4×4 matrix and verify anti-ghosting.
Day 13-14	Audio Output Test	Generating fixed frequency tone using DAC/PWM.
Status:	**30% Complete**	Confirmation of basic input/output functionality.

Week 3: Real-Time Implementation and Polyphony

Date	Status	Observations
Day 15-17	Audio Task	Implementing Audio Task at max priority (constant sample rate).
Day 18-20	Polyphonic Synthesis	Modifying Audio Task for simultaneous notes/chords.
Day 21	Final Assembly	Finalizing soldering of all ≈ 61 keys.
Status:	**70% Complete**	Basic piano functionality is finalized.

5 Software Design (Code Design)

5.1 Detailed Software Architecture

The system relies on a concurrent architecture, likely using **RTIC** or **Embassy** to ensure real-time responsiveness.

Task	Function	Priority	Description
Audio Gen	Audio Output	Max (Critical)	Runs at ≈ 44 kHz, feeding DAC/PWM. Must not be interrupted.
Key Scan	Matrix Input	Medium	Scans matrix ($1-5$ ms) to detect key presses.
Control	Musical Logic	Medium	Calculates Hz and Volume, manages Sustain/Pitch Bend.
Idle	Debugging	Low	Sends debugging messages to PC (RTT).

5.2 Shared Data Structure (Safety in Rust)

To prevent *data races*, all data accessed by both the Audio Task and the I/O Task is protected.

- **Active Notes:** A vector or array (e.g., `Vec<NoteData>`) storing currently active notes.
- **Protection:** Protected by critical sections (RTIC) or channels (Embassy).

5.3 Functional Diagram (Data Flow)

The functional diagram illustrates the path of data from input (keys) to output (sound).

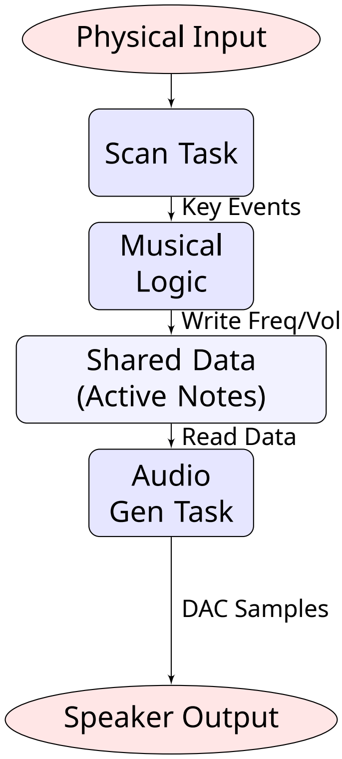

Figure 4: Software Data Flow Diagram

6 Bill of Materials (BOM)

Estimates based on average market rates (RON).

No.	Component	Specifications	Qty	Unit (RON)	Total
		HARDWARE (Core)			
1.	Microcontroller	STM32 Nucleo (F401RE)	1	80.00	80.00
2.	Key Switches	Outemu Linear, 65g	70	1.80	126.00
3.	Diode Kit	1N4148 (20 pcs/kit)	4	12.00	48.00
4.	Audio Amplifier	PAM8403 (2×3W)	1	25.00	25.00
5.	Speaker	4 inch, 8Ω	1	40.00	40.00
6.	Sustain Button	Momentary Pushbutton	1	10.00	10.00
		PROTOTYPING & WIRING			
7.	Protoboard	Breadboard MB102	1	20.00	20.00
8.	Jumper Wires	Male-Male	1 set	15.00	15.00
9.	Matrix Wire	24 AWG Cable	1 set	30.00	30.00
10.	Matrix Conn.	IDC 20 pin	1	10.00	10.00
11.	Speaker Cable	2×0.75 mm^2	1 m	5.00	5.00
		TOOLS			
12.	Soldering Kit	60W Iron + Stand	1 set	125.00	125.00
13.	Resistor Kit	600 pcs	1 set	45.00	45.00
14.	Capacitor Kit	500 pcs	1 set	50.00	50.00
		TOTAL ESTIMATED			$\approx$ **619**

Table 2: Project Bill of Materials