Design and Implementation of a Multi-Instrument Sound Finite State Machine Using PIC18F Microcontroller

Caleb Huddleston & Tioluwanimofe Adesanya

EEC-3220-01 Final Project Report

Trevecca Nazarene University

Nashville, Tennessee, USA

[toadesanya@trevecca.edu](mailto:toadesanya@trevecca.edu) and [cjhuddleston@trevecca.edu](mailto:cjhuddleston@trevecca.edu)

*Abstract*—This paper presents the design and implementation of a complex sound-generating Finite State Machine (FSM) on a PIC18F microcontroller using MPLAB and MCC. The system replicates the sounds of four musical instruments with real-time frequency control using an onboard potentiometer. It also allows users to remember and combine instrument states using external button controls. The sound is generated using the on-chip 8-bit DAC and output to an external audio circuit. This project demonstrates embedded systems design principles, real-time control, and sound waveform synthesis.

Keywords—Final Project, embedded systems, music, synthetic music, Bassoon, Flute, Bassoon, Trumpet, programming, breadboard, DAC, ADC, finite state machine.

1. INTRODUCTION

Finite State Machines (FSMs) are widely used in embedded systems for managing complex state-dependent behavior. In this project, we explore a sound-based FSM application in which a microcontroller is programmed to generate the sounds of multiple musical instruments, with frequency dynamically controlled through analog input. The system responds to multiple user inputs to control instrument selection, memory, and combination, pushing the boundaries of FSM complexity in an embedded audio context.

1. SYSTEM DESIGN AND FUNCTIONALITY

A. System Overview

The FSM system is implemented on a PIC18F microcontroller using MPLAB X IDE with the MCC plugin. The setup includes:

* 3 Instrument-Play Buttons
* 1 Select Button
* 1 Restart Button
* On-chip ADC with onboard potentiometer
* 8-bit DAC output to amplifier and speaker

B. Instrument Selection and Sound Generation

We implemented four instrument waveforms using predefined 64-entry arrays:

* Bassoon
* Flute
* Guitar
* Trumpet

Each waveform simulates a 5-bit audio signal and is output through the speaker at a rate dependent on the potentiometer-controlled frequency (50 Hz to 3.2 kHz).

C. FSM LOGIC AND MEMORY FUNCTIONALITY

* **State 1:** Idle state, no sound unless a button is pressed.
* **State 2:** Active play of one instrument, frequency controlled via ADC.
* **State 3:** Memory mode activated by "Select" button.
* **State 4:** Combined instrument playback up to three remembered instruments.
* **State 5:** Restart clears all memory and halts output.

1. HARDWARE IMPLEMENTATION

A. Port Assignments

TABLE I.

|  |  |
| --- | --- |
| **Function** | **Bit Used** |
| Instrument-Play Button 1 | RD2 |
| Instrument-Play Button 2 | RD3 |
| Instrument-Play Button 3 | RD4 |
| Select Button | RD0 |
| Restart Button | RD1 |
| Potentiometer Input | AN0 |
| DAC Output | RA2 |

B. External Circuitry

The DAC output is passed through the speaker. Button inputs are regulated with simple pullup resistor circuits.

## As seen by the photo of our circuit execution (below),

A circuit board with wires

AI-generated content may be incorrect.

1. PROGRAMMING

C code was developed based on Lab 6 and extended with additional FSM logic. ISR (Interrupt Service Routine) was used to handle button inputs and timer-based DAC updates.

1. RESULTS

The final system successfully generates realistic instrument tones with variable frequency control. Users can mix up to three instruments with retained frequencies and dynamically adjust sound output via the potentiometer.

1. CHALLENGE AND SOLUTION

Sound Mixing and increasing frequency: Achieved by averaging waveform values and normalizing amplitude, but this caused some high noises.

1. CONCLUSION

This project demonstrates the successful application of a finite state machine to manage complex audio behavior on a microcontroller platform. The result is a responsive, memory-capable musical system that highlights embedded control, ADC/DAC use, and user interface design.

ACKNOWLEDGMENT

We would like to give special thanks to Dr. Mark Nardin, who taught us all the concepts we have implemented in this final project, and Dr. Alfred Cawthorne for his assistance in troubleshooting errors.

##### REFERENCES

1. PIC18F47K42 Datasheet, Microchip
2. EEC 3220 Lab Manuals