Technical Module Description: VolumeControl.py

Version: 2.0 Date: January 27, 2025

Authors: Henk Stevens, Olaf Mastenbroek, Onno Janssen

Organization: Stichting Oradio

License: GNU GPL

# 1. Purpose and Overview

The `VolumeControl.py` module is responsible for managing the Oradio device's volume control functionality. It uses an external analog rotary volume control knob connected via an MCP3021 I2C analog-to-digital converter (ADC) to read user input. The script scales this analog input to digital volume levels and adjusts the ALSA mixer accordingly. It provides reliable, real-time volume adjustment and communicates volume change events via a message queue.

# 2. Key Features

* Real-time monitoring of the analog volume knob using MCP3021 I2C ADC.
* Automatic scaling of ADC readings to ALSA digital volume levels.
* Tolerance-based change detection to avoid unnecessary volume adjustments.
* Smooth polling rate adjustment to balance responsiveness and CPU usage.
* Threaded design for non-blocking operation and background monitoring.
* Integration with Oradio's messaging system for asynchronous event notification when volume changes occur.
* Standalone test mode for development, printing ADC readings and volume levels.

# 3. System Components and Dependencies

The volume control system consists of the following components:  
- MCP3021 I2C ADC for reading analog voltage from the volume knob.  
- ALSA sound mixer (`Digital` control) for setting playback volume.  
- Python libraries: `alsaaudio`, `smbus2`, `threading`, and `queue`.

The module interacts with the Oradio message queue to notify other system components when the volume has changed significantly.

# 4. Standalone Use and Testing

The `VolumeControl.py` module includes a built-in standalone test mode that allows developers to test the volume control independently of the full Oradio system. This mode helps verify the correct operation of the ADC reading, volume scaling, and ALSA mixer adjustments. The test mode also demonstrates the responsiveness of the polling and change detection logic.

## Volume adjustment

The settings of the minimum and maximum olume can be tuned via the:

# Raw volume units

VOLUME\_MINIMUM = 105. # the minimum volume (in dB relative)

VOLUME\_MAXIMUM = 200 # the maximum volume (in dB relative)

These are defined in oradio\_const.py. It defines the 8 bit volume control of the TAS5756 amplifier and act as a master volume.

## How to Run in Standalone Test Mode

To run the module in standalone test mode, execute the script directly from the terminal:  
  
```bash  
python3 VolumeControl.py  
```  
The script initializes the ADC monitoring and ALSA mixer, then continuously reads the volume knob position. It prints the ADC values and corresponding volume settings to the console. Use Ctrl+C to safely stop the test.

## Expected Test Output

While running the test, you should observe output similar to the following when the knob is adjusted:  
- `ADC Value: 512, Volume: 65`  
- `ADC Value: 845, Volume: 90`  
- `Volume set to: 65` (log output from ALSA volume setting)  
These outputs confirm that the ADC is being read correctly and that the volume is being scaled and applied as expected.

## Integration with Message Queue

When used as part of the Oradio system, the module integrates with a shared message queue. Every significant change in volume triggers a message of the following format:  
  
```python  
{'type': MESSAGE\_TYPE\_VOLUME, 'state': MESSAGE\_STATE\_CHANGED, 'error': MESSAGE\_NO\_ERROR}  
```  
Other system components can listen to this message queue to respond to volume adjustments (e.g., updating status displays or adjusting related audio streams).