**Oradio3 System Installation Script**

**1. Introduction**

The **Oradio3 system installation script** is a shell script designed to automate the setup process for a **Raspberry Pi running Debian Bookworm (64-bit)**. Its primary objective is to convert the Raspberry Pi into a fully configured **Oradio3 media player** capable of streaming audio, interfacing with external hardware (such as USB drives and I2C devices), and supporting **Spotify Connect** via **Librespot**. The script performs a series of tasks, including installing system packages, configuring services, setting up peripheral interfaces, and ensuring the system can autonomously boot and run the Oradio3 media player.

This installation process is highly tailored to Oradio3's requirements, ensuring that the Raspberry Pi is ready to serve as an audio streaming device with minimal manual intervention.

**2. Script Workflow**

The script follows a structured, sequential flow to configure the system. It is divided into multiple stages to ensure each component is installed, configured, and validated correctly. Below is an overview of the primary stages and their associated actions:

1. **OS Compatibility Check**
   * The script begins by verifying the operating system version using the lsb\_release command to ensure it is running **Debian GNU/Linux 12 (Bookworm)** 64-bit. If the OS does not match this requirement, the script exits with an error message.
2. **Directory Setup**
   * The script creates necessary directories for logs, Python files, Spotify resources, and installation files. This step ensures that all paths are organized and available for subsequent configurations. The following directories are created:
     + logging (to store log files)
     + Python (for Python scripts)
     + Spotify (for Spotify-related resources)
     + install\_resources (for configuration files)
3. **Package Management**
   * The script installs a series of required packages for the system. These packages are necessary for Oradio3's operation and include:
     + System packages like jq, python3-dev, libasound2-dev, mpd, and mpc.
     + Python development packages like python-mpd2, requests, smbus2, and rpi-lgpio.
   * The installation ensures that any missing or outdated packages are resolved automatically.
4. **Python Environment Setup**
   * The script checks for the existence of a Python virtual environment. If none exists, it creates one at the user’s home directory (~/.venv). The script then configures the environment to automatically activate upon login by modifying the .bashrc file.
   * Additionally, the script installs necessary Python modules for Oradio3's functionality, including packages for handling MPD (Music Player Daemon), network connectivity, and hardware interactions.
5. **System Configuration**
   * **Host Name and Networking**: The script changes the hostname to oradio and updates /etc/hosts and /etc/hostname accordingly. It also configures mDNS (Multicast DNS) so that the device can be accessed via the hostname oradio.local on the local network.
   * **Wi-Fi Country Setting**: The script sets the Wi-Fi country to NL (Netherlands) to ensure the device can connect to Wi-Fi networks correctly, respecting local regulations.

**3. Hardware Configuration**

Oradio3 requires certain hardware configurations, particularly for peripheral devices such as USB drives and I2C components. The script automates the configuration of these peripherals, ensuring that Oradio3 works seamlessly with connected hardware.

1. **USB Device Configuration**
   * The script installs and configures a custom script (usb-mount.sh) to automatically mount USB storage devices when detected. It ensures that any connected USB drive is correctly mounted to /media on the system.
   * **USB Quirks**: The script adds a specific quirk (usb-storage.quirks=0781:5583:u) to the kernel command line to address issues with some USB storage devices, ensuring compatibility.
   * **Service Setup**: The script installs a systemd service (usb-mount@.service) to manage USB device mounting at boot time. The service handles adding and removing devices as needed.
   * **Log File Monitoring**: Logs related to USB device mounting and potential errors are captured in a dedicated log file (usb.log). The script checks this log to report any mounting issues.
2. **I2C Interface Setup**
   * The script enables the I2C interface, which is necessary for communicating with I2C-based hardware devices such as sensors or displays.
   * The script installs the required I2C kernel modules and ensures the system is configured to automatically load them at boot.
3. **Audio Configuration**
   * **ALSA Equalizer**: The script configures the audio system to use an ALSA equalizer (alsaequal.bin). This allows fine-grained control over the audio output, ensuring high-quality sound tailored to the user’s preferences.
   * **SoftVol Configuration**: The script sets up multiple virtual sound cards (SoftVol) to handle different audio outputs such as SoftVolSpotCon1, SoftVolSysSound, and SoftVolMPD. This enables flexible audio routing for various use cases (e.g., MPD playback, system sound).
   * **Speaker Test and Volume Settings**: The script performs an initial speaker test on each virtual sound card and adjusts the volume levels to a predefined setting for consistency.
4. **Backlighting Configuration**
   * The script configures a systemd service to control the backlighting of the Raspberry Pi's display, adjusting brightness levels based on user activity or system conditions.

**4. Service Configuration and Spotify Integration**

Oradio3 includes several services that ensure the system operates as a fully functional media player, capable of both local and network-based audio playback.

1. **MPD Service**
   * The script installs and configures the **Music Player Daemon (MPD)**, a server for playing audio files stored on the device or accessible over the network. It ensures the MPD service is set to start automatically upon boot.
2. **Spotify Connect Integration**
   * The script integrates **Librespot**, an open-source implementation of Spotify Connect, allowing users to stream music from Spotify to the Oradio3 device.
   * **Raspotify Installation**: Initially, the script installs raspotify, which includes librespot, and then disables the raspotify service to avoid conflict. Instead, it configures a dedicated systemd service to run **Librespot** independently.
   * **Spotify Event Handler**: The script also installs a custom **Spotify event handler** to manage and respond to Spotify-related events (e.g., when playback starts or stops).
3. **Logging and System Monitoring**
   * **Log Rotation**: To prevent log files from growing too large and overwhelming the system, the script installs a **logrotate** configuration that ensures logs are rotated regularly.
   * Logs are stored for various system components, including:
     + USB device events (usb.log).
     + Spotify-related events (spotify.log).
     + General installation and configuration steps (install.log).

**5. Final System Configuration**

Once the main components are installed and configured, the script proceeds with additional system configurations to ensure everything runs smoothly after the next reboot.

1. **Autostart and Reboot Handling**
   * If the installation requires a reboot to finalize certain changes (e.g., USB configuration), the script ensures that the installation continues automatically upon reboot by modifying the user’s .bashrc file. The reboot process is initiated, and the system will resume the installation steps once the system restarts.
   * After the reboot, the script restores the .bashrc file to its original state and completes the remaining configuration steps.
2. **Final System Configurations**
   * **Wi-Fi Country and Hostname Settings**: The script sets the system’s Wi-Fi country and ensures the hostname is set to oradio. The hostname is also updated in /etc/hosts to reflect the local network name oradio.local.
   * **Serial Number Display**: The script configures the Raspberry Pi to display its serial number and Oradio3 software version upon login. This information is retrieved from the Raspberry Pi's OTP (One-Time Programmable) memory.
   * **Log File for Software Version**: The script logs the current **Oradio3 software version** and **Raspberry Pi serial number** in /var/log/oradio\_sw\_version.log for future reference.
3. **Reboot to Finalize Installation**
   * Once all configurations are complete, the script initiates a final reboot to ensure that all services are running correctly and the system is ready to be used as an Oradio3 media player.

**6. Error Handling and Logging**

The script includes robust error handling and logging mechanisms throughout the installation process. Each significant step, whether it involves package installation, service configuration, or hardware setup, is logged with detailed information. If any step fails, the script reports the issue in the log files, providing useful feedback for troubleshooting.

Key error logs are captured in:

* **usb.log**: Tracks USB mounting issues.
* **spotify.log**: Captures Spotify-related events.
* **install.log**: Logs general installation progress and issues.
* **traceback.log**: Captures any unexpected errors or exceptions during execution.

By inspecting these log files, users can identify any issues that may have occurred during installation or configuration and take corrective actions.

**Conclusion**

The **Oradio3 system installation script** is a comprehensive solution for setting up a Raspberry Pi as an Oradio3 media player. By automating the process of installing system packages, configuring hardware, and setting up Spotify Connect functionality, the script reduces the complexity of deploying Or

adio3. The script’s modular design ensures flexibility, allowing for customization and expansion as needed. It also incorporates robust error handling and logging, making it easy to troubleshoot and maintain the system after installation.