Firmware versioning is an essential part of the firmware development process, allowing for easy identification of firmware releases and facilitating updates, debugging, and maintenance. Here's a simple overview of how firmware versioning can be implemented:

**1. Decide on a Versioning Scheme:**

Common versioning schemes include:

Semantic Versioning (SemVer): This is of the form MAJOR.MINOR.PATCH:

MAJOR version for incompatible API changes.

MINOR version for added features in a backward-compatible manner.

PATCH version for backward-compatible bug fixes.

Example: 1.3.5

Date-Based Versioning: Using the release date as a version, like YYYY.MM.DD.

Example: 2023.10.10

Sequential Versioning: Simply increasing a number for each release. Example: v1, v2, v3...

**2. Embed the Version Information:**

The firmware version information should be embedded in the firmware binary, typically as string literals or constants, which can be accessed via an API or command. This allows devices to report their firmware version, aiding in diagnostics and update checks.

**3. Store & Track Versions in Source Control:**

Every firmware release should be tagged in your source control system (e.g., using git tag in Git). This makes it easy to identify code associated with a particular release and aids in managing patches or updates.

**4. Maintain a Changelog:**

For each new version, maintain a log detailing:

Date of the release.

New features added.

Bugs fixed.

Any backward-incompatible changes.

**5. Consistency:**

Whatever scheme you decide, it's essential to remain consistent. Inconsistent versioning can lead to confusion, particularly if deploying OTA updates or managing a large fleet of devices.

**6. Device Compatibility Check:**

When designing an OTA system or firmware update mechanism, always ensure the new firmware version is compatible with the device before initiating the update. This can be done by checking device model numbers, hardware revisions, and current firmware versions.

**7. Alpha/Beta Tags:**

For firmware in testing or pre-release, use tags like alpha or beta (e.g., 1.3.5-alpha) to differentiate from stable releases.

**8. Backup & Archive:**

Always keep a backup of old firmware binaries and their source code. If a new version introduces unforeseen issues, it might be necessary to rollback devices to a previous stable version.

In essence, a well-structured firmware versioning system helps in managing and tracking firmware releases, ensuring that devices receive the correct updates, and aids developers in diagnosing issues.