Comprehensive Microcontroller Refactor Plan

Version: 2.0 (Aligned with ADR-20 v2.5 and Master Project Plan `master_project_plan_v1` Phases 2 & 3)
Date: 2025-05-22

Overall Goal: Refactor and unify firmware for Controllers C1, C2, and C3 to implement ADR-10 MQTT payloads, achieve SSOT-driven configuration (including pins and MQTT topics derived from `autogen_config.h` generated by `build.py`), enhance robustness with a common C++ framework and FSM utilities, and implement persistent error logging for restart reasons (including no-publish timeouts). This plan depends on and incorporates decisions from **ADR-20** and assumes completion of relevant tasks in `ssot_detailed_plan_v2_0` (especially `autogen_config.h` generation).

Status Legend:

- ☐ `[ ]` To Do
- `[/]` In Progress
- `[X]` Done
- `[B]` Blocked

Phase 0: Foundational Setup & Definitions (Conceptual)

Goal: Define common structures and interfaces for the microcontroller firmware. Much of the configuration definition and `autogen_config.h` generation is now handled by `ssot_detailed_plan_v2_0`.

- `[X]` **P0.1: Define Core FSM States Enumeration**
  - *Details:* Common `enum State` defined: `[ WAIT, SETUP_HW, CONNECT_WIFI, SYNC_NTP, CONNECT_MQTT, PROCESS_COMMANDS, READ_SENSORS, PUBLISH_DATA, RESTART]` . Each controller implements relevant states.
- `[X]` **P0.2: Define Microcontroller Restart Reason Enumeration**
  - *Details:* Common `enum RestartReason` created (e.g., `UNKNOWN_ERROR`, `WIFI_CONNECTION_FAILED`, `NO_PUBLISH_TIMEOUT`).
- `[X]` **P0.3: Define `RestartReasonLogger` Interface**
  - *Details:* Public methods and EEPROM strategy conceptually defined. Implementation in Phase 1 (Task P1.5).
- `N/A` **P0.3.A: Define/Update Pydantic Models for Microcontroller Configurations**
  - *Note:* This is now handled by `ssot_detailed_plan_v2_0` **Task 1.3** (Finalize Pydantic Models in `component_configs.py`). The `MicrocontrollerConfig` YAML is an input to `build.py`.
- `[X]` **P0.4: Define `autogen_config.h` Target Structure (Generated by `build.py`)**
  - *Details:* The target structure for per-controller `autogen_config.h` files is specified in **ADR-**

20, Section II.C.

- **Crucial Note:** `autogen_config.h` is **generated by** `build.py` (as per `ssot_detailed_plan_v2_0` Task 3.4). Any initial manual creation of this file is for structural understanding *only*. Firmware development **MUST** use the `build.py` generated version.
- Content includes: `DEVICE_ID`, WiFi/MQTT/NTP configs, and Hardware Point Definitions:
  - `#define POINT_NAME_[LocalName]`
  - `#define UUID_[LocalName]_[ASPECT]`
  - `#define PIN_[LocalName]`
  - `#define MODE_[LocalName]`
  - `#define INITIAL_STATE_[LocalName]`
  - `#define TOPIC_[LocalName]_[ASPECT]` (Full MQTT topics derived by `build.py` per **ADR-20, Section II.C**, e.g., `mush/c2/actuators/fruiting_chamber_heating_pad/readback`, `mush/temperature_driver_fruiting/commands/c2/fruiting_chamber_heating_pad/write`)
  - Critical timing constants (e.g., `MAX_TIME_NO_PUBLISH_MS`).

- `[X]` **P0.5: Define** `MonitoredPublishPoint` **Base Class Interface**
  - *Details:* Simple base class with `virtual const char* getPointName() = 0;` (returns `POINT_NAME_...` from `autogen_config.h`) and `unsigned long lastPublishTime = 0;`.

Phase 1: Develop & Test Common Firmware Library (Corresponds to Master Plan Phase 2)

- `[X]` **P1.1: Develop** `JsonBuilder` **Service**
  - *Details:* Adheres to ADR-10. Uses `ArduinoJson`. Unit tested.
- `[X]` **P1.2: Develop** `NtpService` **Service**
  - *Details:* Includes retry logic. Unit tested.
- `[X]` **P1.3: Develop** `MqttService` **Service**
  - *Details:* Implements LWT, reconnection, message callback, and command queueing (ADR-16). Unit tested.

☐ `[ ]` **P1.4: Develop Minimal FSM Utilities (** `FsmUtils` **)**

- ☐ `[ ]` Design and implement `FsmUtils.h/.cpp` in `common_firmware_lib`.
- ☐ `[ ]` Ensure common state enum (P0.1) is accessible.
- ☐ `[ ]` Implement `const char* stateToString(State state);`.
- ☐ `[ ]` Implement `void transitionToState(State &currentStateVariable, State newState, unsigned long &stateEntryTimeMillis);` .
- ☐ `[ ]` Implement `bool checkTimeout(unsigned long stateEntryTimeMillis, unsigned long timeoutDurationMillis);` .
- ☐ `[ ]` Unit test these utilities.

- [ ] **[ ] P1.5: Implement `RestartReasonLogger.h/.cpp` in `common_firmware_lib` (was P4.1)**
  - [ ] [ ] Implement methods defined in P0.3 using EEPROM.
  - [ ] [ ] Ensure robust EEPROM operations.
  - [ ] [ ] Unit test `RestartReasonLogger`.

Phase 2: Refactor Controller C2 (Actuators) (Corresponds to Master Plan Task 3.1)

Goal: Update C2 to use the common library, FSM structure, and SSOT-driven configuration via `build.py`-generated `autogen_config.h`.

- **Pre-requisite:** C2's `MicrocontrollerConfig` YAML defined (as per `ssot_detailed_plan_v2_0` Task 2.4) and `autogen_config_c2.h` correctly generated by `build.py` (as per `ssot_detailed_plan_v2_0` Task 3.4).

- [ ] **[ ] P2.C2.1: Integrate `build.py` Generated `autogen_config_c2.h` for C2**
  - [ ] [ ] Ensure C2 firmware includes and uses the `autogen_config_c2.h` generated by `build.py`.
- [ ] **[ ] P2.C2.2: Implement `main.cpp` for C2 using FSM Pattern & Common Libraries**
  - [ ] [ ] Structure `main.cpp` with FSM switch, using `FsmUtils`.
  - [ ] [ ] Integrate and use `NtpService`, `MqttService`, and `JsonBuilder`.
  - [ ] [ ] Source all configurations (WiFi, MQTT, `DEVICE_ID`, NTP, all MQTT topics via `TOPIC_...` macros, pin definitions via `PIN_...` macros, timeouts like `MAX_TIME_NO_PUBLISH_MS`) **exclusively** from `autogen_config_c2.h`.
  - [ ] [ ] Use `PinControl` with `PIN_...` and `MODE_...` defines. Initialize pins using `INITIAL_STATE_...` defines.
  - [ ] [ ] Implement relevant FSM states for C2 (e.g., `SETUP_HW`, `CONNECT_WIFI`, `SYNC_NTP`, `CONNECT_MQTT`, `PROCESS_COMMANDS` subscribing to `TOPIC_..._WRITE` from `autogen_config_c2.h`, `PUBLISH_DATA` for actuator readbacks publishing to `TOPIC_..._READBACK`, `WAIT`, `RESTART`).
- [ ] **[ ] P2.C2.3: Implement ADR-10 Payloads for C2 Data (Actuator Readbacks)**
  - [ ] [ ] If C2 publishes actuator readbacks, use `JsonBuilder` and `MqttService::publishJson()` in `PUBLISH_DATA` state, using `TOPIC_..._READBACK` defines from `autogen_config_c2.h`.
- [ ] **[ ] P2.C2.4: Create Actuator Data Wrappers for C2 (Monitored Points, if applicable)**
  - [ ] [ ] If C2 publishes actuator readbacks needing freshness monitoring, create wrappers inheriting from `MonitoredPublishPoint`. `getPointName()` returns `POINT_NAME_...` from `autogen_config_c2.h`.
  - [ ] [ ] Populate and manage `monitored_points` vector and `lastPublishTime` updates.
- [ ] **[ ] P2.C2.5: Integrate `RestartReasonLogger` and No-Publish Timeout in C2**
  - [ ] [ ] Instantiate and setup `RestartReasonLogger`.

- [ ] [ ] In relevant FSM states (on failure), call `restartLogger.storeRestartReason()` before transitioning to `RESTART`.
- [ ] [ ] In `WAIT` state, periodically check `monitored_points` against `MAX_TIME_NO_PUBLISH_MS` (from `autogen_config_c2.h`). If timeout, log `NO_PUBLISH_TIMEOUT` and restart.

- [ ] [ ] **P2.C2.6: Test Controller C2 Thoroughly**
  - [ ] [ ] Verify WiFi, NTP, MQTT connectivity.
  - [ ] [ ] Verify actuator control via MQTT commands (to `TOPIC_..._WRITE`) and readback publication (from `TOPIC_..._READBACK`).
  - [ ] [ ] Test FSM state transitions, error handling, restart reasons, and no-publish timeouts.

Phase 3: Refactor Controller C1 (Sensors) (Corresponds to Master Plan Task 3.2)

Goal: Update C1 similarly, using `build.py`-generated `autogen_config.h`.

- **Pre-requisite:** C1's `MicrocontrollerConfig` YAML defined (as per `ssot_detailed_plan_v2_0` Task 2.4) and `autogen_config_c1.h` correctly generated by `build.py` (as per `ssot_detailed_plan_v2_0` Task 3.4).

- [ ] [ ] **P3.C1.1: Integrate `build.py` Generated `autogen_config_c1.h` for C1**
  - [ ] [ ] Ensure C1 firmware includes and uses the `autogen_config_c1.h` generated by `build.py`.
- [ ] [ ] **P3.C1.2: Implement `main.cpp` for C1 using FSM Pattern & Common Libraries**
  - [ ] [ ] Structure `main.cpp` with FSM switch, using `FsmUtils`.
  - [ ] [ ] Integrate `NtpService`, `MqttService`, `JsonBuilder`.
  - [ ] [ ] Source all configurations **exclusively** from `autogen_config_c1.h`.
  - [ ] [ ] Implement relevant FSM states for C1 (e.g., `SETUP_HW`, `CONNECT_WIFI`, `SYNC_NTP`, `CONNECT_MQTT`, `READ_SENSORS`, `PUBLISH_DATA` to `TOPIC_..._DATA`, `WAIT`, `RESTART`).
- [ ] [ ] **P3.C1.3: Implement ADR-10 Payloads for C1 Sensor Data**
  - [ ] [ ] In `PUBLISH_DATA` state, use `JsonBuilder` for sensor data.
  - [ ] [ ] Use `MqttService::publishJson()` to send to `TOPIC_..._DATA` defines from `autogen_config_c1.h`.
- [ ] [ ] **P3.C1.4: Create Sensor Data Wrappers for C1 (Monitored Points)**
  - [ ] [ ] For each data point, create wrappers inheriting from `MonitoredPublishPoint`. `getPointName()` returns `POINT_NAME_...` from `autogen_config_c1.h`.
  - [ ] [ ] Manage `monitored_points` vector and `lastPublishTime`.
- [ ] [ ] **P3.C1.5: Integrate `RestartReasonLogger` and No-Publish Timeout in C1**
  - [ ] [ ] Integrate `RestartReasonLogger`.
  - [ ] [ ] Implement and test no-publish timeout logic for C1's `monitored_points` using `MAX_TIME_NO_PUBLISH_MS` from `autogen_config_c1.h`.

- [ ] `[ ]` **P3.C1.6: Test Controller C1 Thoroughly**
  - [ ] `[ ]` Verify connectivity, sensor data reading, ADR-10 publication.
  - [ ] `[ ]` Test FSM, error handling, restart reasons, no-publish timeouts.

Phase 4: Refactor Controller C3 (SCD41 Sensor) (Corresponds to Master Plan Task 3.3)

Goal: Update C3 similarly, using `build.py`-generated `autogen_config.h`.

- **Pre-requisite:** C3's `MicrocontrollerConfig` YAML defined (as per `ssot_detailed_plan_v2_0` Task 2.4) and `autogen_config_c3.h` correctly generated by `build.py` (as per `ssot_detailed_plan_v2_0` Task 3.4).

- [ ] `[ ]` **P4.C3.1: Integrate `build.py` Generated `autogen_config_c3.h` for C3**
  - [ ] `[ ]` Ensure C3 firmware includes and uses the `autogen_config_c3.h` generated by `build.py`.
- [ ] `[ ]` **P4.C3.2: Implement `main.cpp` for C3 using FSM Pattern & Common Libraries**
  - [ ] `[ ]` Structure `main.cpp` with FSM switch, using `FsmUtils`.
  - [ ] `[ ]` Integrate common library services.
  - [ ] `[ ]` Source all configurations **exclusively** from `autogen_config_c3.h`.
  - [ ] `[ ]` Implement SCD41 sensor interaction logic.
  - [ ] `[ ]` Implement relevant FSM states for C3.
- [ ] `[ ]` **P4.C3.3: Implement ADR-10 Payloads for C3 Sensor Data**
  - [ ] `[ ]` In `PUBLISH_DATA` state, use `JsonBuilder` for SCD41 readings (CO2, Temp, Humidity) into separate payloads.
  - [ ] `[ ]` Use `MqttService::publishJson()` to send to respective `TOPIC_..._DATA` defines from `autogen_config_c3.h`.
- [ ] `[ ]` **P4.C3.4: Create SCD41 Data Wrapper for C3 (Monitored Points)**
  - [ ] `[ ]` Create wrappers for CO2, Temp, Humidity data points inheriting from `MonitoredPublishPoint`. `getPointName()` returns `POINT_NAME_...`.
  - [ ] `[ ]` Manage `monitored_points` vector and `lastPublishTime`.
- [ ] `[ ]` **P4.C3.5: Integrate `RestartReasonLogger` and No-Publish Timeout in C3**
  - [ ] `[ ]` Integrate `RestartReasonLogger`.
  - [ ] `[ ]` Implement and test no-publish timeout for SCD41 data points using `MAX_TIME_NO_PUBLISH_MS` from `autogen_config_c3.h`.
- [ ] `[ ]` **P4.C3.6: Test Controller C3 Thoroughly**
  - [ ] `[ ]` Verify connectivity, SCD41 readings, ADR-10 publication.
  - [ ] `[ ]` Test FSM, error handling, restart reasons, no-publish timeouts.

Phase 5: Final Review, Testing & Documentation Update (Corresponds to

Master Plan Phase 5)

Goal: Ensure overall system stability, consistency, and documentation for the microcontroller aspects.

- [ ] `[ ]` **P5.1: Code Review All Controllers & Shared Library**
  - [ ] `[ ]` Perform thorough code review of `common_firmware_lib` and `main.cpp` for C1, C2, C3.
  - [ ] `[ ]` Check for consistency, adherence to ADR-20 (via `autogen_config.h`), error handling.
- [ ] `[ ]` **P5.2: System-Level Integration Testing (Microcontroller Focus)**
  - [ ] `[ ]` Run all three controllers simultaneously.
  - [ ] `[ ]` Verify correct data flow to MQTT broker (topics and payloads as per ADR-20 and ADR-10).
  - [ ] `[ ]` Test various operational scenarios and induced error conditions.
- [ ] `[ ]` **P5.3: Update Microcontroller-Specific Documentation**
  - [ ] `[ ]` Update READMEs for C1, C2, C3 and `common_firmware_lib`.
  - [ ] `[ ]` Ensure any uC-specific details in `V2 Architecture and Overview.pdf` are accurate.
- `N/A` **P5.4: `build.py` Automation for `autogen_config.h`**
  - *Note:* This is fully covered in `ssot_detailed_plan_v2_0` **Tasks 3.4 and 3.5**. This microcontroller plan *consumes* the artifacts generated by `build.py`.