UAV Flight Controller Specifications Document

Tanner Chase
Christian Hall
Anthony Wood

November 1, 2025

Signature Page

Signature	*Signature*	*Signature*
Date and email	*Date and email*	*Date and email*

Revision History

Revision	Description	Author	Date	Approval
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				

Contents

Specifications

1 SCOPE

(a) **General:**

(b) **Acronyms:**

2 APPLICABLE DOCUMENTS

The following documents shown shall form part of the specifications for this project. In the event of a conflict between requirements, priority shall first go to the contract, second to this document, and lastly to these reference documents.

(a) **Government Documents** *This is where to put MIL-Specs, MIL-STDs, NASA specs and so forth. Be sure to include the revision level and date.*

(b) **Industry Documents** *This is where to put ANSI, ASTM, ASME, IEEE, Company specifications and so forth. Both this section and government documents can be divided up into logical subcategories.*

3 STAKEHOLDER REQUIREMENTS

4 ENGINEERING REQUIREMENTS

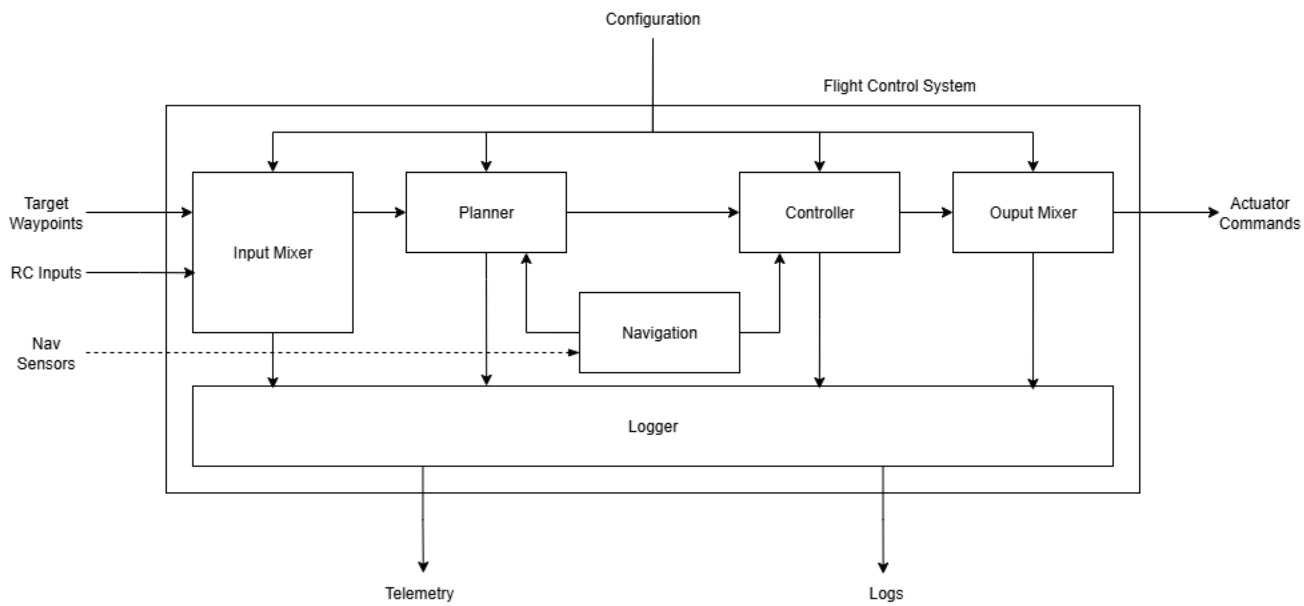

Figure 1. System functional diagram

4.1 System Requirements

The System provides the hardware and software foundation for all onboard modules. It includes processing, hardware capabilities, interfaces, hardware abstraction, and power management. The System is comprised of five main parts: Processing, Built in HW Capabilities, HW Interfaces, HAL, and Power Management.

4.1.1 Interfaces

4.1.1.1 Inputs

4.1.1.1.1 Power supply, configuration parameters, and external signals

4.1.1.2 Outputs

4.1.1.2.1 Processed data, actuator signals, and telemetry

4.1.1.3 Internal

4.1.1.3.1 Internal buses, clocks, and control signals

4.1.2 Processing

The Processing part provides the computational resources and pipeline for all onboard software modules.

4.1.2.1 Provide sufficient compute for all modules.

4.1.2.2 Support real-time scheduling and isolation.

4.1.2.3 Ensure fault tolerance and recovery.

4.1.3 Built in HW Capabilities

Built-in hardware capabilities include sensors, actuators, and onboard peripherals required for mission execution.

4.1.3.1 Provide required sensors and actuators.

4.1.3.2 Ensure hardware health monitoring.

4.1.3.3 Support hardware expansion where feasible.

4.1.4 HW Interfaces

Hardware interfaces provide connectivity between onboard modules and external devices.

4.1.4.1 Support standard communication protocols.

4.1.4.2 Ensure robust electrical and logical connections.

4.1.4.3 Provide diagnostics for interface health.

4.1.5 HAL

The Hardware Abstraction Layer (HAL) provides a uniform interface to hardware resources, abstracting device specifics from higher-level modules.

4.1.5.1 Expose consistent APIs for hardware access.

4.1.5.2 Isolate hardware changes from application logic.

4.1.5.3 Support safe fallback on hardware faults.

4.1.6 Power Management

Power Management ensures efficient and reliable power distribution to all system components.

4.1.6.1 Monitor and regulate power supply.

4.1.6.2 Support low-power and standby modes.

4.1.6.3 Protect against overcurrent and undervoltage.

4.1.7 Sub-Parts

The System is comprised of four functional sub-parts:

4.1.7.1 Planner

4.1.7.2 Controller

4.1.7.3 Navigation

4.1.7.4 Logger

4.2 Planner

The purpose of the Planner is to generate a series of realizable states for the UAV. The planner allows for both autonomous and pilot-directed operation. Inputs from both autonomous and pilot-directed modes are evaluated and potentially adjusted to fit safety and feasibility constraints. The inputs are evaluated based on the current UAV state and the current tolerances from configuration parameters. The Planner is comprised of three submodules: Waypoint Planner, RC Mixer, and State Select. Each has aids in the overall functionality of the Planner.

4.2.1 Interfaces

4.2.1.1 Inputs

4.2.1.1.1 Waypoints and configuration parameters

4.2.1.2 Outputs

4.2.1.2.1 Planned waypoints

4.2.1.3 Internal

4.2.1.3.1 Blah

4.2.2 Waypoint Planner

The Waypoint Planner submodule is responsible for generating feasible waypoint trajectories based on input commands and the current UAV state. It processes both autonomous and pilot-directed inputs, ensuring that the generated waypoints adhere to safety and feasibility constraints.

4.2.2.1 Consume waypoints and configuration parameters as defined in XXX:location

4.2.2.2 Output Planned waypoints as defined in XXX:location

4.2.2.3 Execute within allocated CPU and power budgets.

4.2.2.4 Maintain performance across expected environmental conditions.

4.2.2.5 Ensure generated waypoints adhere to safety constraints.

4.2.2.6 Modular design with clear interfaces.

4.2.2.7 Follow applicable software development standards.

4.2.3 RC Mixer

Description

4.2.3.1 Consume Item as defined in XXX:location

4.2.3.2 Output Item as defined in XXX:location

4.2.3.3 Execute within allocated CPU and power budgets.

4.2.3.4 Maintain performance across expected environmental conditions.

4.2.3.5 Ensure generated waypoints adhere to safety constraints.

4.2.3.6 Modular design with clear interfaces.

4.2.3.7 Follow applicable software development standards.

4.2.4 State Select

Description

4.2.4.1 Consume Item as defined in XXX:location

4.2.4.2 Output Item as defined in XXX:location

4.2.4.3 Execute within allocated CPU and power budgets.

4.2.4.4 Maintain performance across expected environmental conditions.

4.2.4.5 Ensure generated waypoints adhere to safety constraints.

4.2.4.6 Modular design with clear interfaces.

4.2.4.7 Follow applicable software development standards.

4.3 Controller

The Controller computes actuator setpoints from planned trajectories and state estimates. It implements required control algorithms and scheduling, ensuring safe and efficient operation of the UAV actuators. The Controller is comprised of four submodules: Preprocessor, PVA Controller, ATL Controller, and Control Distributor. Each submodule contributes to translating planned trajectories into actuator commands.

4.3.1 Interfaces

4.3.1.1 Inputs

4.3.1.1.1 Planned trajectories and state estimates

4.3.1.2 Outputs

4.3.1.2.1 Actuator setpoints

4.3.1.3 Internal

4.3.1.3.1 Internal control signals

4.3.2 Preprocessor

The Preprocessor submodule is responsible for preparing and conditioning inputs for the control loops, including filtering and validation of sensor data before it is used by the controllers.

4.3.2.1 Consume raw sensor measurements as defined in XXX:location

4.3.2.2 Output filtered sensor streams as defined in XXX:location

4.3.2.3 Execute within allocated CPU and power budgets.

4.3.2.4 Maintain performance across expected environmental conditions.

4.3.2.5 Filtering shall never mask critical fault indicators.

4.3.2.6 Modular design with clear interfaces.

4.3.2.7 Follow applicable software development standards.

4.3.3 PVA Controller

The PVA (Position/Velocity/Acceleration) Controller submodule computes control setpoints to track planned trajectories and maintain the UAV along the desired path.

4.3.3.1 Consume waypoints and state estimates as defined in XXX:location

4.3.3.2 Output actuator-level setpoints as defined in XXX:location

4.3.3.3 Execute within allocated CPU and power budgets.

4.3.3.4 Maintain performance across expected environmental conditions.

4.3.3.5 Ensure bounded errors and safe fallback when estimates are invalid.

4.3.3.6 Modular design with clear interfaces.

4.3.3.7 Follow applicable software development standards.

4.3.4 ATL Controller

The ATL (Axis/Attitude/Torque-Level) Controller submodule translates higher-level commands into low-level actuator demands, respecting actuator limits and constraints.

4.3.4.1 Consume desired attitude and torque references as defined in XXX:location

4.3.4.2 Output low-level actuator commands as defined in XXX:location

4.3.4.3 Execute within allocated CPU and power budgets.

4.3.4.4 Maintain performance across expected environmental conditions.

4.3.4.5 Enforce actuator limits and safe modes under faults.

4.3.4.6 Modular design with clear interfaces.

4.3.4.7 Follow applicable software development standards.

4.3.5 Control Distributor

The Control Distributor submodule routes control outputs to the correct actuator channels, mapping logical control channels to physical hardware outputs.

4.3.5.1 Consume controller output commands as defined in XXX:location

4.3.5.2 Output hardware actuator signals as defined in XXX:location

4.3.5.3 Execute within allocated CPU and power budgets.

4.3.5.4 Maintain performance across expected environmental conditions.

4.3.5.5 Prevent misrouting; provide diagnostics and traceability.

4.3.5.6 Modular design with clear interfaces.

4.3.5.7 Follow applicable software development standards.

4.4 Navigation

The Navigation module provides accurate state estimation and sensor selection to support planning and control by running estimators, selecting appropriate sensors, and publishing state estimates. The Navigation module is comprised of two submodules: Sensor Selector and Estimator. Each submodule contributes to robust and accurate state estimation from available sensor inputs.

4.4.1 Interfaces

4.4.1.1 Inputs

4.4.1.1.1 Sensor measurements and configuration parameters

4.4.1.2 Outputs

4.4.1.2.1 State estimates and validity flags

4.4.1.3 Internal

4.4.1.3.1 Internal sensor selection signals

4.4.2 Sensor Selector

The Sensor Selector submodule evaluates available sensor inputs and selects the best sensors for the current conditions, ensuring robust state estimation even when some sensors fail or degrade.

4.4.2.1 Provide prioritized sensor streams and fallback choices.

4.4.2.2 Consume all available sensor feeds and health indicators as defined in XXX:location

4.4.2.3 Output selected sensor feeds for estimator as defined in XXX:location

4.4.2.4 Execute within allocated CPU and power budgets.

4.4.2.5 Maintain performance across expected environmental conditions.

4.4.2.6 Ensure safe fallbacks when preferred sensors are unavailable.

4.4.2.7 Modular design with clear interfaces.

4.4.2.8 Follow applicable software development standards.

4.4.3 Estimator

The Estimator submodule fuses selected sensor inputs using sensor fusion algorithms to produce the best estimate of the system state (position, velocity, attitude, etc.).

4.4.3.1 Provide pose, velocity and other required state variables with uncertainty metrics.

4.4.3.2 Consume selected sensor streams and configuration as defined in XXX:location

4.4.3.3 Output state estimates with timestamps as defined in XXX:location

4.4.3.4 Execute within allocated CPU and power budgets.

4.4.3.5 Maintain performance across expected environmental conditions.

4.4.3.6 Provide safe outputs when inputs are inconsistent or stale.

4.4.3.7 Modular design with clear interfaces.

4.4.3.8 Follow applicable software development standards.

4.5 Logger

The Logger records mission data and provides mechanisms for transmission or storage by collecting telemetry, writing to storage, and/or sending data via communications links. The Logger is comprised of two submodules: Storage and Transmitt. Each submodule handles data persistence and transmission for mission telemetry and diagnostic information.

4.5.1 Interfaces

4.5.1.1 Inputs

4.5.1.1.1 Telemetry streams and configuration parameters

4.5.1.2 Outputs

4.5.1.2.1 Stored logs and transmitted telemetry packets

4.5.1.3 Internal

4.5.1.3.1 Internal logging control signals

4.5.2 Storage

The Storage submodule is responsible for persisting telemetry and logs to onboard non-volatile memory, ensuring data is retained for later retrieval and analysis.

4.5.2.1 Support retrieval by timestamp and event markers.

4.5.2.2 Consume telemetry streams to be persisted, storage configuration as defined in XXX:location

4.5.2.3 Output stored log files or records retrievable by time/index as defined in XXX:location

4.5.2.4 Execute within allocated CPU and power budgets.

4.5.2.5 Maintain performance across expected environmental conditions.

4.5.2.6 Protect critical logs and prevent data loss on faults.

4.5.2.7 Modular design with clear interfaces.

4.5.2.8 Follow applicable software development standards.

4.5.3 Transmitt

The Transmitt submodule handles transmission of selected telemetry data to ground stations or other external systems via available communications links.

4.5.3.1 Provide telemetry prioritization and retransmission.

4.5.3.2 Consume selected telemetry streams and transmission policy as defined in XXX:location

4.5.3.3 Output telemetry packets transmitted via communications link as defined in XXX:location

4.5.3.4 Execute within allocated CPU and power budgets.

4.5.3.5 Maintain performance across expected environmental conditions.

4.5.3.6 Prioritize safety/health telemetry under degraded links.

4.5.3.7 Modular design with clear interfaces.

4.5.3.8 Follow applicable software development standards.

5 VERIFICATION OF REQUIREMENTS

Possible verification methods include:

5.1 Inspection:

Inspection is a method of verification consisting of investigation, without the use of special laboratory appliances or procedures, to determine compliance with requirements. Inspection is generally nondestructive and includes (but is not limited to) visual examination, manipulation, gauging, and measurement.

5.2 Demonstration:

Demonstration is a method of verification that is limited to readily observable functional operation to determine compliance with requirements. This method shall not require the use of special equipment or sophisticated instrumentation.

5.3 Analysis:

Analysis is a method of verification, taking the form of the processing of accumulated results and conclusions, intended to provide proof that verification of a requirement has been accomplished. The analytical results may be based on engineering study, compilation or interpretation of existing information, similarity to previously verified requirements, or derived from lower level examinations, tests, demonstrations, or analyses.

5.4 Direct Test:

Test is a method of verification that employs technical means, including (but not limited to) the evaluation of functional characteristics by use of special equipment or instrumentation, simulation techniques, and the application of established principles and procedures to determine compliance with requirements.

5.1 Verify Coverage of Stakeholder Requirements

Paragraph Number	Test Type	Tester's Name	Pass/Fail	Date