Student Specifications Document Template Guide

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

1 SCOPE **10**

2 APPLICABLE DOCUMENTS **11**

3 STAKEHOLDER REQUIREMENTS **12**

4 ENGINEERING REQUIREMENTS **13**

4.1 System Requirements 14
4.1.1 Purpose 14
4.1.2 Operation 14
4.1.3 Functional Requirements 14
4.1.4 Interfaces 14
4.1.5 Non-Functional Requirements 14
4.2 Processing Core 16
4.2.1 Purpose 16
4.2.2 Operation 16
4.2.3 Functional Requirements 16
4.2.4 Interfaces 16
4.2.5 Non-Functional Requirements 16
4.3 Planner 18
4.3.1 Functional Requirements 18
4.3.2 Interfaces 18
4.3.3 Non-Functional Requirements 18
4.4 Waypoint Planner 20
4.4.1 Purpose 20
4.4.2 Operation 20
4.4.3 Functional Requirements 20
4.4.4 Interfaces 20
4.4.5 Non-Functional Requirements 20
4.5 RC Mixer 21
4.5.1 Purpose 21
4.5.2 Operation 21
4.5.3 Functional Requirements 21
4.5.4 Interfaces 21
4.5.5 Non-Functional Requirements 21
4.6 State Select 22

4.6.1 Purpose . . . 22
4.6.2 Operation . . . 22
4.6.3 Functional Requirements . . . 22
4.6.4 Interfaces . . . 22
4.6.5 Non-Functional Requirements . . . 22
4.7 Controller . . . 23
4.7.1 Purpose . . . 23
4.7.2 Operation . . . 23
4.7.3 Functional Requirements . . . 23
4.7.4 Interfaces . . . 23
4.7.5 Non-Functional Requirements . . . 23
4.8 Preprocessor . . . 25
4.8.1 Purpose . . . 25
4.8.2 Operation . . . 25
4.8.3 Functional Requirements . . . 25
4.8.4 Interfaces . . . 25
4.8.5 Non-Functional Requirements . . . 25
4.9 PVA Controller . . . 26
4.9.1 Purpose . . . 26
4.9.2 Operation . . . 26
4.9.3 Functional Requirements . . . 26
4.9.4 Interfaces . . . 26
4.9.5 Non-Functional Requirements . . . 26
4.10 ATL Controller . . . 27
4.10.1 Purpose . . . 27
4.10.2 Operation . . . 27
4.10.3 Functional Requirements . . . 27
4.10.4 Interfaces . . . 27
4.10.5 Non-Functional Requirements . . . 27
4.11 Control Distributor . . . 28
4.11.1 Purpose . . . 28
4.11.2 Operation . . . 28
4.11.3 Functional Requirements . . . 28
4.11.4 Interfaces . . . 28
4.11.5 Non-Functional Requirements . . . 28
4.12 Navigation . . . 29
4.12.1 Purpose . . . 29

4.12.2 Operation . . . 29
4.12.3 Functional Requirements . . . 29
4.12.4 Interfaces . . . 29
4.12.5 Non-Functional Requirements . . . 29
4.13 Sensor Selector . . . 31
4.13.1 Purpose . . . 31
4.13.2 Operation . . . 31
4.13.3 Functional Requirements . . . 31
4.13.4 Interfaces . . . 31
4.13.5 Non-Functional Requirements . . . 31
4.14 Estimator . . . 32
4.14.1 Purpose . . . 32
4.14.2 Operation . . . 32
4.14.3 Functional Requirements . . . 32
4.14.4 Interfaces . . . 32
4.14.5 Non-Functional Requirements . . . 32
4.15 Logger . . . 33
4.15.1 Purpose . . . 33
4.15.2 Operation . . . 33
4.15.3 Functional Requirements . . . 33
4.15.4 Interfaces . . . 33
4.15.5 Non-Functional Requirements . . . 33
4.16 Storage . . . 35
4.16.1 Purpose . . . 35
4.16.2 Operation . . . 35
4.16.3 Functional Requirements . . . 35
4.16.4 Interfaces . . . 35
4.16.5 Non-Functional Requirements . . . 35
4.17 Transmitt . . . 36
4.17.1 Purpose . . . 36
4.17.2 Operation . . . 36
4.17.3 Functional Requirements . . . 36
4.17.4 Interfaces . . . 36
4.17.5 Non-Functional Requirements . . . 36
4.18 Processing Architecture . . . 37
4.18.1 Purpose . . . 37
4.18.2 Operation . . . 37

4.18.3 Functional Requirements . . . 37
4.18.4 Interfaces . . . 37
4.18.5 Non-Functional Requirements . . . 37
4.19 RTOS . . . 39
4.19.1 Purpose . . . 39
4.19.2 Operation . . . 39
4.19.3 Functional Requirements . . . 39
4.19.4 Interfaces . . . 39
4.19.5 Non-Functional Requirements . . . 39
4.20 HAL . . . 40
4.20.1 Purpose . . . 40
4.20.2 Operation . . . 40
4.20.3 Functional Requirements . . . 40
4.20.4 Interfaces . . . 40
4.20.5 Non-Functional Requirements . . . 40
4.21 Hardware Platform . . . 41
4.21.1 Purpose . . . 41
4.21.2 Operation . . . 41
4.21.3 Functional Requirements . . . 41
4.21.4 Interfaces . . . 41
4.21.5 Non-Functional Requirements . . . 41
4.22 Microprocessor . . . 43
4.22.1 Purpose . . . 43
4.22.2 Operation . . . 43
4.22.3 Functional Requirements . . . 43
4.22.4 Interfaces . . . 43
4.22.5 Non-Functional Requirements . . . 43
4.23 Hardware I/O ports . . . 44
4.23.1 Purpose . . . 44
4.23.2 Operation . . . 44
4.23.3 Functional Requirements . . . 44
4.23.4 Interfaces . . . 44
4.23.5 Non-Functional Requirements . . . 44
4.24 Power Supply . . . 45
4.24.1 Purpose . . . 45
4.24.2 Operation . . . 45
4.24.3 Functional Requirements . . . 45

4.24.4 Interfaces 45
4.24.5 Non-Functional Requirements 45
4.25 Configuration Tool 46
4.25.1 Purpose 46
4.25.2 Operation 46
4.25.3 Functional Requirements 46
4.25.4 Interfaces 46
4.25.5 Non-Functional Requirements 46

5 VERIFICATION OF REQUIREMENTS 48
5.1 Verify Coverage of Stakeholder Requirements 49

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

4.1 System Requirements

4.1.1 Purpose

Purpose

4.1.2 Operation

Operation

4.1.3 Functional Requirements

(a) Requirement 1

4.1.3.1 Submodules

(a) Processing Core

(b) Processing Architecture

(c) Hardware Platform

(d) Configuration Tool

4.1.4 Interfaces

4.1.4.1 Inputs

(a) Input 1

4.1.4.2 Outputs

(a) Output 1

4.1.5 Non-Functional Requirements

4.1.5.1 Energy

(a) Energy Requirement 1

4.1.5.2 Environments

(a) Environment Requirement 1

4.1.5.3 Safety

(a) Safety Requirement 1

4.1.5.4 Structure

(a) Structure Requirement 1

4.1.5.5 Standards and Regulations

(a) Standards and Regulations Requirement 1

4.2 Processing Core

4.2.1 Purpose

Purpose of the Processing Core: provide the main computation and coordination services for planning, control, navigation, and logging.

4.2.2 Operation

Operation of the Processing Core: receives sensor inputs, runs planning and control algorithms, and issues actuator commands and logs.

4.2.3 Functional Requirements

(a) The Processing Core shall host the Planner, Controller, Navigation, and Logger submodules and provide inter-module communication.

(b) The Processing Core shall provide scheduling and communication primitives used by its submodules.

4.2.3.1 Submodules

(a) Planner

(b) Controller

(c) Navigation

(d) Logger

4.2.4 Interfaces

4.2.4.1 Inputs

(a) Sensor data and configuration parameters

4.2.4.2 Outputs

(a) Actuator commands, telemetry, and stored logs

4.2.5 Non-Functional Requirements

4.2.5.1 Energy

(a) The Processing Core shall operate within the power budget allocated for the onboard processor.

4.2.5.2 Environments

(a) The Processing Core shall tolerate expected operating temperatures and vibration levels.

4.2.5.3 Safety

(a) The Processing Core shall not issue unsafe actuator commands under fault conditions.

4.2.5.4 Structure

(a) The Processing Core shall expose well-defined APIs for its submodules.

4.2.5.5 Standards and Regulations

(a) The Processing Core shall follow applicable coding and safety guidelines.

4.3 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.

4.3.1 Functional Requirements

(a) Make da plans

4.3.1.1 Submodules

(a) Waypoint Planner

(b) RC Mixer

(c) State Select

4.3.2 Interfaces

4.3.2.1 Inputs

(a) Waypoints and configuration parameters

4.3.2.2 Outputs

(a) Planned waypoints

4.3.3 Non-Functional Requirements

4.3.3.1 Energy

(a) The Planner shall operate within the power budget allocated for the onboard processor.

4.3.3.2 Environments

(a) The Planner shall tolerate expected operating temperatures and vibration levels.

4.3.3.3 Safety

(a) The Planner shall not issue unsafe actuator commands under fault conditions.

4.3.3.4 Structure

(a) The Planner shall expose well-defined APIs for its submodules.

4.3.3.5 Standards and Regulations

(a) The Planner shall follow applicable coding and safety guidelines.

4.4 Waypoint Planner

4.4.1 Purpose

Convert high-level paths into waypoint lists suitable for controllers.

4.4.2 Operation

Consume planned paths and output discrete waypoints with timing and tolerances.

4.4.3 Functional Requirements

(a) Produce waypoint lists with time or speed annotations.

4.4.4 Interfaces

4.4.4.1 Inputs

(a) Planned path segments

4.4.4.2 Outputs

(a) Discrete waypoint messages

4.4.5 Non-Functional Requirements

4.4.5.1 Energy

(a) Operate within allocated compute and power budgets for planning tasks.

4.4.5.2 Environments

(a) Maintain deterministic behavior across expected operating temperatures and vibration.

4.4.5.3 Safety

(a) Generate waypoint sets that do not violate safety constraints or no-go zones.

4.4.5.4 Structure

(a) Expose versioned interfaces for waypoint messages and configuration.

4.4.5.5 Standards and Regulations

(a) Adhere to applicable data and coding standards for planning algorithms.

4.5 RC Mixer

4.5.1 Purpose

Merge pilot inputs with autonomous commands where applicable.

4.5.2 Operation

Accept RC inputs and planned commands, output blended commands respecting control logic.

4.5.3 Functional Requirements

(a) Support priority rules between pilot and autonomous commands.

4.5.4 Interfaces

4.5.4.1 Inputs

(a) RC stick commands, autonomous command stream

4.5.4.2 Outputs

(a) Mixed command stream to controllers

4.5.5 Non-Functional Requirements

4.5.5.1 Energy

(a) Operate within the control loop power and CPU budgets.

4.5.5.2 Environments

(a) Maintain performance under expected EMI and temperature ranges.

4.5.5.3 Safety

(a) Enforce priority and safety interlocks to prevent unsafe command blending.

4.5.5.4 Structure

(a) Provide traceable mapping between RC inputs and mixed outputs.

4.5.5.5 Standards and Regulations

(a) Follow applicable safety and control interface standards.

4.6 State Select

4.6.1 Purpose

Choose higher-level operating states (e.g., manual, assisted, autonomous).

4.6.2 Operation

Monitor inputs and select current operational mode, publishing mode state to other modules.

4.6.3 Functional Requirements

(a) Provide deterministic state transitions with guards and safety checks.

4.6.4 Interfaces

4.6.4.1 Inputs

(a) Pilot commands, system health, mission status

4.6.4.2 Outputs

(a) Current operating state

4.6.5 Non-Functional Requirements

4.6.5.1 Energy

(a) Minimal additional power and CPU overhead per transition decision.

4.6.5.2 Environments

(a) Robust to sensor dropouts and environmental disturbances.

4.6.5.3 Safety

(a) Provide auditable and recoverable transitions with fail-safe defaults.

4.6.5.4 Structure

(a) Documented state machine with clear guards and actions.

4.6.5.5 Standards and Regulations

(a) Conform to applicable safety/state-machine design practices.

4.7 Controller

4.7.1 Purpose

Compute actuator setpoints from planned trajectories and state estimates.

4.7.2 Operation

Run control loops (e.g., PVA, ATL) and distribute control outputs to actuators.

4.7.3 Functional Requirements

(a) Implement required control algorithms and scheduling.

4.7.3.1 Submodules

(a) Preprocessor

(b) PVA Controller

(c) ATL Controller

(d) Control Distributor

4.7.4 Interfaces

4.7.4.1 Inputs

(a) Waypoints, state estimates, preprocessor outputs

4.7.4.2 Outputs

(a) Actuator commands

4.7.5 Non-Functional Requirements

4.7.5.1 Energy

(a) Execute within the control loop CPU and power budgets.

4.7.5.2 Environments

(a) Maintain stability across environmental conditions and sensor noise.

4.7.5.3 Safety

(a) Ensure bounded outputs and safe behavior under saturation and faults.

4.7.5.4 Structure

(a) Modular controller design with clear scheduling and interfaces.

4.7.5.5 Standards and Regulations

(a) Follow applicable control and software standards.

4.8 Preprocessor

4.8.1 Purpose

Prepare inputs for control loops (filtering, safety checks).

4.8.2 Operation

Filter and condition raw inputs before they are used by the controller.

4.8.3 Functional Requirements

(a) Provide filtered and validated sensor data to controllers.

4.8.4 Interfaces

4.8.4.1 Inputs

(a) Raw sensor measurements

4.8.4.2 Outputs

(a) Filtered sensor streams

4.8.5 Non-Functional Requirements

4.8.5.1 Energy

(a) Processing overhead shall fit within the loop budget.

4.8.5.2 Environments

(a) Filters shall maintain performance under expected sensor disturbances.

4.8.5.3 Safety

(a) Filtering shall never mask critical fault indicators.

4.8.5.4 Structure

(a) Configurable filter parameters with documented effects.

4.8.5.5 Standards and Regulations

(a) Follow applicable signal-processing guidelines.

4.9 PVA Controller

4.9.1 Purpose

Position/velocity/acceleration controller for trajectory following.

4.9.2 Operation

Compute setpoints to track waypoints and trajectories.

4.9.3 Functional Requirements

(a) Maintain tracking error within specified bounds.

4.9.4 Interfaces

4.9.4.1 Inputs

(a) Waypoints and state estimates

4.9.4.2 Outputs

(a) Actuator-level setpoints

4.9.5 Non-Functional Requirements

4.9.5.1 Energy

(a) Execute within allocated CPU/power for PVA computations.

4.9.5.2 Environments

(a) Maintain stability across expected disturbances and noise.

4.9.5.3 Safety

(a) Ensure bounded errors and safe fallback when estimates are invalid.

4.9.5.4 Structure

(a) Parameterized gains with traceable tuning records.

4.9.5.5 Standards and Regulations

(a) Follow applicable control-system standards.

4.10 ATL Controller

4.10.1 Purpose

Axis/attitude/torque-level controller.

4.10.2 Operation

Translate higher-level commands to low-level actuator demands.

4.10.3 Functional Requirements

(a) Provide actuator commands that respect actuator limits.

4.10.4 Interfaces

4.10.4.1 Inputs

(a) Desired attitude and torque references

4.10.4.2 Outputs

(a) Low-level actuator commands

4.10.5 Non-Functional Requirements

4.10.5.1 Energy

(a) Operate within low-level loop budgets.

4.10.5.2 Environments

(a) Maintain performance across expected thermal and EMI conditions.

4.10.5.3 Safety

(a) Enforce actuator limits and safe modes under faults.

4.10.5.4 Structure

(a) Clear interfaces to actuator drivers and health monitors.

4.10.5.5 Standards and Regulations

(a) Adhere to applicable safety and hardware interfacing standards.

4.11 Control Distributor

4.11.1 Purpose

Route control outputs to the correct actuator channels.

4.11.2 Operation

Distribute controller outputs to physical I/O interfaces.

4.11.3 Functional Requirements

(a) Map logical control channels to hardware outputs.

4.11.4 Interfaces

4.11.4.1 Inputs

(a) Controller output commands

4.11.4.2 Outputs

(a) Hardware actuator signals

4.11.5 Non-Functional Requirements

4.11.5.1 Energy

(a) Minimal overhead in routing commands to outputs.

4.11.5.2 Environments

(a) Maintain timing integrity under expected conditions.

4.11.5.3 Safety

(a) Prevent misrouting; provide diagnostics and traceability.

4.11.5.4 Structure

(a) Configurable mapping from control channels to hardware outputs.

4.11.5.5 Standards and Regulations

(a) Follow applicable avionics/robotics interfacing standards where relevant.

4.12 Navigation

4.12.1 Purpose

Provide accurate state estimation and sensor selection to support planning and control.

4.12.2 Operation

Run estimators, select appropriate sensors, and publish state estimates.

4.12.3 Functional Requirements

(a) Provide timely, bounded-uncertainty state estimates.

4.12.3.1 Submodules

(a) Sensor Selector

(b) Estimator

4.12.4 Interfaces

4.12.4.1 Inputs

(a) Raw sensor data and configuration

4.12.4.2 Outputs

(a) State estimate messages

4.12.5 Non-Functional Requirements

4.12.5.1 Energy

(a) Complete estimation within allocated compute budget.

4.12.5.2 Environments

(a) Maintain accuracy across sensor noise and environmental changes.

4.12.5.3 Safety

(a) Provide validity flags and fail-safe estimates upon degraded inputs.

4.12.5.4 Structure

(a) Provide uncertainty metrics and timestamps with each estimate.

4.12.5.5 Standards and Regulations

(a) Follow applicable estimation and data handling standards.

4.13 Sensor Selector

4.13.1 Purpose

Choose the best sensor inputs for the current conditions.

4.13.2 Operation

Evaluate sensor health and availability, output selected sensor streams.

4.13.3 Functional Requirements

(a) Provide prioritized sensor streams and fallback choices.

4.13.4 Interfaces

4.13.4.1 Inputs

(a) All available sensor feeds and health indicators

4.13.4.2 Outputs

(a) Selected sensor feeds for estimator

4.13.5 Non-Functional Requirements

4.13.5.1 Energy

(a) Selection logic shall be lightweight and efficient.

4.13.5.2 Environments

(a) Robust to sensor dropouts and environmental disturbances.

4.13.5.3 Safety

(a) Ensure safe fallbacks when preferred sensors are unavailable.

4.13.5.4 Structure

(a) Clearly defined selection criteria and priorities.

4.13.5.5 Standards and Regulations

(a) Follow applicable data quality and redundancy practices.

4.14 Estimator

4.14.1 Purpose

Fuse selected sensor inputs into a state estimate.

4.14.2 Operation

Run sensor fusion algorithms and output best-guess system state.

4.14.3 Functional Requirements

(a) Provide pose, velocity and other required state variables with uncertainty metrics.

4.14.4 Interfaces

4.14.4.1 Inputs

(a) Selected sensor streams and configuration

4.14.4.2 Outputs

(a) State estimates with timestamps

4.14.5 Non-Functional Requirements

4.14.5.1 Energy

(a) Execute within compute and power budgets for estimation.

4.14.5.2 Environments

(a) Maintain estimation quality across expected operating conditions.

4.14.5.3 Safety

(a) Provide safe outputs when inputs are inconsistent or stale.

4.14.5.4 Structure

(a) Provide consistent time alignment and data schemas.

4.14.5.5 Standards and Regulations

(a) Comply with relevant estimation and logging standards.

4.15 Logger

4.15.1 Purpose

Record mission data and provide mechanisms for transmission or storage.

4.15.2 Operation

Collect telemetry, write to storage, and/or send data via communications links.

4.15.3 Functional Requirements

(a) Preserve critical telemetry and provide retrieval interfaces.

4.15.3.1 Submodules

(a) Storage

(b) Transmitt

4.15.4 Interfaces

4.15.4.1 Inputs

(a) Telemetry streams from other modules

4.15.4.2 Outputs

(a) Stored logs and outgoing telemetry

4.15.5 Non-Functional Requirements

4.15.5.1 Energy

(a) Logging shall not exceed allocated power/CPU budgets.

4.15.5.2 Environments

(a) Maintain logging integrity across operating conditions.

4.15.5.3 Safety

(a) Logging should not interfere with real-time control operations.

4.15.5.4 Structure

(a) Provide schemas and retention policies for stored data.

4.15.5.5 Standards and Regulations

(a) Follow applicable data retention and privacy regulations.

4.16 Storage

4.16.1 Purpose

Persist logs for later retrieval.

4.16.2 Operation

Write telemetry to onboard non-volatile memory with indexing.

4.16.3 Functional Requirements

(a) Support retrieval by timestamp and event markers.

4.16.4 Interfaces

4.16.4.1 Inputs

(a) Telemetry streams to be persisted, storage configuration

4.16.4.2 Outputs

(a) Stored log files or records retrievable by time/index

4.16.5 Non-Functional Requirements

4.16.5.1 Energy

(a) Operate within storage device power and bandwidth limits.

4.16.5.2 Environments

(a) Ensure data integrity across temperature/vibration ranges.

4.16.5.3 Safety

(a) Protect critical logs and prevent data loss on faults.

4.16.5.4 Structure

(a) Provide indexing and retrieval mechanisms (by time/index).

4.16.5.5 Standards and Regulations

(a) Follow data integrity and storage standards.

4.17 Transmitt

4.17.1 Purpose

Transmit selected telemetry offboard.

4.17.2 Operation

Send compressed telemetry or health messages through the communications stack.

4.17.3 Functional Requirements

(a) Provide telemetry prioritization and retransmission.

4.17.4 Interfaces

4.17.4.1 Inputs

(a) Selected telemetry streams and transmission policy

4.17.4.2 Outputs

(a) Telemetry packets transmitted via communications link

4.17.5 Non-Functional Requirements

4.17.5.1 Energy

(a) Respect power constraints for the communications subsystem.

4.17.5.2 Environments

(a) Maintain link robustness across expected RF/EMI conditions.

4.17.5.3 Safety

(a) Prioritize safety/health telemetry under degraded links.

4.17.5.4 Structure

(a) Provide message schemas and prioritization policies.

4.17.5.5 Standards and Regulations

(a) Follow spectrum usage and data regulations.

4.18 Processing Architecture

4.18.1 Purpose

Purpose: define OS and HAL abstractions to support real-time processing.

4.18.2 Operation

Provides scheduling, task isolation, and hardware abstraction layers.

4.18.3 Functional Requirements

(a) Support RTOS features and a hardware abstraction layer for portable drivers.

4.18.3.1 Submodules

(a) RTOS

(b) HAL

4.18.4 Interfaces

4.18.4.1 Inputs

(a) Driver requests and configuration

4.18.4.2 Outputs

(a) Timing and scheduling primitives to tasks

4.18.5 Non-Functional Requirements

4.18.5.1 Energy

(a) Support power-aware scheduling and low-power modes in RTOS and drivers where applicable.

4.18.5.2 Environments

(a) Maintain timing determinism and reliability across expected thermal, vibration, and EMI environments.

4.18.5.3 Safety

(a) Provide task isolation, memory protection, and watchdog integration for safety-critical functions.

4.18.5.4 Structure

(a) Expose stable, versioned APIs for scheduling and hardware abstraction with clear dependency boundaries.

4.18.5.5 Standards and Regulations

(a) Conform to relevant real-time and coding standards (e.g., MISRA/CERT where applicable) and any platform certification requirements.

4.19 RTOS

4.19.1 Purpose

Purpose: real-time operating system services.

4.19.2 Operation

Task scheduling, priority inversion handling, timing services.

4.19.3 Functional Requirements

(a) Provide deterministic scheduling for control tasks.

4.19.4 Interfaces

4.19.4.1 Inputs

(a) Task creation requests, timing constraints, and system configuration

4.19.4.2 Outputs

(a) Scheduling decisions, timers, and inter-task signaling primitives

4.19.5 Non-Functional Requirements

4.19.5.1 Energy

(a) Provide power-aware scheduling where applicable.

4.19.5.2 Environments

(a) Maintain timing guarantees under expected operating conditions.

4.19.5.3 Safety

(a) Support priority inversion mitigation and safety-critical task isolation.

4.19.5.4 Structure

(a) Provide documented APIs for tasks, timers, and synchronization.

4.19.5.5 Standards and Regulations

(a) Conform to relevant RTOS certification or best-practice standards.

4.20 HAL

4.20.1 Purpose

Purpose: hardware abstraction for portable drivers.

4.20.2 Operation

Provide uniform APIs for peripherals and I/O across platforms.

4.20.3 Functional Requirements

(a) Expose safe, well-documented interfaces for hardware access.

4.20.4 Interfaces

4.20.4.1 Inputs

(a) Driver requests from higher layers, hardware configuration

4.20.4.2 Outputs

(a) Standardized peripheral access APIs and events

4.20.5 Non-Functional Requirements

4.20.5.1 Energy

(a) Support low-power modes and efficient peripheral usage.

4.20.5.2 Environments

(a) Operate reliably across expected electrical and thermal conditions.

4.20.5.3 Safety

(a) Ensure safe access to hardware, enforcing limits and protections.

4.20.5.4 Structure

(a) Maintain stable, versioned APIs across platforms.

4.20.5.5 Standards and Regulations

(a) Comply with applicable hardware interface standards.

4.21 Hardware Platform

4.21.1 Purpose

Purpose: physical computing and power platform that hosts the processing and I/O.

4.21.2 Operation

Hosts the microprocessor, I/O ports, and provides power to subsystems.

4.21.3 Functional Requirements

(a) Meet mechanical, thermal, and electrical constraints for deployment.

4.21.3.1 Submodules

(a) Microprocessor

(b) Hardware I/O ports

(c) Power Supply

4.21.4 Interfaces

4.21.4.1 Inputs

(a) Power and environmental inputs

4.21.4.2 Outputs

(a) Power rails and I/O signals

4.21.5 Non-Functional Requirements

4.21.5.1 Energy

(a) Meet system power budgets with efficient conversion and distribution; support low-power modes where required.

4.21.5.2 Environments

(a) Operate within specified temperature, shock, and vibration profiles; provide EMI/ESD robustness.

4.21.5.3 Safety

(a) Include electrical protections (fusing, OVP/UVP/OCP) and safe power-down behavior; isolate fault domains.

4.21.5.4 Structure

(a) Provide mechanical mounting, connectorization, and labeling suitable for serviceability; document interfaces and constraints.

4.21.5.5 Standards and Regulations

(a) Comply with applicable electrical safety and EMC standards for the deployment environment.

4.22 Microprocessor

4.22.1 Purpose

Purpose: execute software for all processing tasks.

4.22.2 Operation

Run RTOS and application software within the hardware constraints.

4.22.3 Functional Requirements

(a) Provide sufficient CPU, memory, and peripherals to host the software stack.

4.22.4 Interfaces

4.22.4.1 Inputs

(a) Power and clock inputs, boot configuration

4.22.4.2 Outputs

(a) Processing capability exposed via buses and I/O interfaces

4.22.5 Non-Functional Requirements

4.22.5.1 Energy

(a) Meet power consumption targets including low-power modes.

4.22.5.2 Environments

(a) Operate within specified temperature and vibration ratings.

4.22.5.3 Safety

(a) Support watchdogs and fault detection mechanisms.

4.22.5.4 Structure

(a) Provide documentation for cores, memory maps, and peripherals.

4.22.5.5 Standards and Regulations

(a) Comply with relevant processor and EMC standards.

4.23 Hardware I/O ports

4.23.1 Purpose

Purpose: expose physical interfaces to sensors and actuators.

4.23.2 Operation

Provide GPIO, serial, analog, and bus interfaces for peripherals.

4.23.3 Functional Requirements

(a) Meet signaling and timing requirements of connected devices.

4.23.4 Interfaces

4.23.4.1 Inputs

(a) Sensor and actuator connections, configuration inputs

4.23.4.2 Outputs

(a) Electrical signals to external devices (GPIO, serial, analog, buses)

4.23.5 Non-Functional Requirements

4.23.5.1 Energy

(a) Respect current and power limits for all ports.

4.23.5.2 Environments

(a) Meet environmental tolerances and EMI/ESD protections.

4.23.5.3 Safety

(a) Provide protections against shorts and over-voltage conditions.

4.23.5.4 Structure

(a) Clearly document pinouts and electrical characteristics.

4.23.5.5 Standards and Regulations

(a) Comply with relevant electrical and interface standards.

4.24 Power Supply

4.24.1 Purpose

Purpose: supply stable, filtered power to the system.

4.24.2 Operation

Regulate and distribute power, monitor health, and provide protections.

4.24.3 Functional Requirements

(a) Provide required voltages and current with redundancy as specified.

4.24.4 Interfaces

4.24.4.1 Inputs

(a) Input power source(s), control signals for power management

4.24.4.2 Outputs

(a) Regulated power rails and status/health signals

4.24.5 Non-Functional Requirements

4.24.5.1 Energy

(a) Efficiency shall meet specified targets across load ranges.

4.24.5.2 Environments

(a) Operate across specified thermal and vibration conditions.

4.24.5.3 Safety

(a) Include protections (OVP, UVP, OCP) and safe shutdown behavior.

4.24.5.4 Structure

(a) Provide diagnostic and status reporting for power rails.

4.24.5.5 Standards and Regulations

(a) Comply with relevant electrical safety and EMC standards.

4.25 Configuration Tool

4.25.1 Purpose

Purpose: provide an interface for configuring system parameters and mission plans.

4.25.2 Operation

Run on a host or ground-station, produce configuration files or commands for the system.

4.25.3 Functional Requirements

(a) Allow editing and validation of configuration parameters and mission plans.

4.25.4 Interfaces

4.25.4.1 Inputs

(a) User inputs and configuration files

4.25.4.2 Outputs

(a) Validated configuration data for the system

4.25.5 Non-Functional Requirements

4.25.5.1 Energy

(a) Operate efficiently on supported host systems, with modest CPU/memory usage during typical editing and validation.

4.25.5.2 Environments

(a) Support specified OS/platform versions; operate reliably offline and under intermittent filesystem or network conditions.

4.25.5.3 Safety

(a) Prevent creation of unsafe configurations via validation, schemas, and guardrails; provide role-appropriate access controls if applicable.

4.25.5.4 Structure

(a) Version configuration schemas; support migration tooling, backups, and audit/change tracking.

4.25.5.5 Standards and Regulations

(a) Adhere to relevant data handling/privacy policies and file format standards required by the system.

5 VERIFICATION OF REQUIREMENTS

Possible verification methods include:

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.

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.

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.

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