**GNSS Resilience**

**System Test Plan**

**Version 0.10**

**12/5/2024**

# Document Control

## Distribution List

The following list of people will receive a copy of this document every time an updated version of this document becomes available:

Teaching assistants:

Pate, Clay

Customer(s): Clifford, Jayson F.

Neighbors, Jake C.

Project team members:

Arena, Cristina M.

Cappers, Haskell A.

Legro, Jonathan

Lofton, Tyler R.

## Change Summary

The following table details changes made between versions of this document:

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Modifier** | **Description** |
| 0.1 | 12/2/2024 | Tyler Lofton | Initial creation, including the distribution list, Introduction, and document structure. |
| 0.2 | 12/2/2024 | Jonathan Legro | Updated the Introduction, System Overview, Testing Approach, and aligned content with the current development phase. |
| 0.3 | 12/3/2024 | Jonathan Legro | Expanded on the Introduction to include a Document Overview and References. |
| 0.4 | 12/3/2024 | Jonathan Legro | Incorporated both hardware and software components to the Test Plan and Test Cases. |
| 0.5 | 12/4/2024 | Jonathan Legro | Added future feature and performance tests. Updated environment, assumptions, and risks. Simplified criteria. |
| 0.6 | 12/5/2024 | Cristina Arena | Added test cases for hardware components. |
| 0.7 | 12/5/2024 | Jonathan Legro | Added a hardware test case following the same step-by-step style as the software test cases. |
| 0.8 | 12/5/2024 | Jonathan Legro | Updated the software and hardware test plan tables in Section 2 for consistency and clarity. |
| 0.9 | 12/5/2024 | Jonathan Legro | Improved the formatting and style of test cases (e.g., TC-SW-005) for readability and easier maintenance. |
| 0.10 | 12/5/2024 | Jonathan Legro | Ensured that both software and hardware test cases clearly reference their corresponding requirements. |

Table of Contents

[Document Control 1](#_Toc501055487)

[Distribution List 2](#_Toc2140320239)

[Change Summary 2](#_Toc814700039)

[1. Introduction 3](#_Toc1343051910)

[1.1. Purpose 4](#_Toc1958450673)

[1.2. Scope 4](#_Toc572943230)

[1.3. System Overview 4](#_Toc1420976978)

[1.4. Testing Approach Overview 4](#_Toc945884535)

[1.5. Testing Entrance Criteria 5](#_Toc2087864072)

[1.6. Document Overview 5](#_Toc223800036)

[1.7. References 5](#_Toc1605260678)

[2. Testing Approach 5](#_Toc458807635)

[2.1. Testing Types 5](#_Toc1282508880)

[2.1.1. Stability Testing 6](#_Toc585366195)

[2.1.2. Interface Testing 6](#_Toc1420612405)

[2.1.3. Functionality Testing 6](#_Toc1602137936)

[2.1.4. Preparatory Functional Testing (Future Phases) 6](#_Toc2594769)

[2.1.5. Performance and Stress Testing (Future Phases) 6](#_Toc1700814107)

[2.2. Testing Suspension Criteria and Resumption Requirements 7](#_Toc506908795)

[2.2.1. Suspension Criteria 7](#_Toc1507684405)

[2.2.2. Resumption Requirements 7](#_Toc1195198852)

[2.3. Test Management Requirements 7](#_Toc269049897)

[2.4. Personnel Requirements 7](#_Toc1374808224)

[2.5. Testing Environment 8](#_Toc95688615)

[2.6. Testing Assumptions 8](#_Toc45920734)

[2.7. Testing Risks and Contingencies 8](#_Toc11728882)

[2.8. Cost 8](#_Toc15342146)

[2.9. Test Plan 8](#_Toc1693073449)

[2.9.1. Software 8](#_Toc1791698269)

[2.9.2. Hardware 9](#_Toc90641152)

[3. Test Schedule 10](#_Toc1898714784)

[4. Traceability Matrix and Defect Tracking 10](#_Toc1660399656)

[4.1. Traceability Matrix 10](#_Toc480457635)

[4.2. Defect Severity Definitions 11](#_Toc1467108806)

[5. Test Cases 11](#_Toc512606034)

[5.1. Software 12](#_Toc1283240004)

[5.1.1. Test Case SW- 1 12](#_Toc789374881)

[5.1.2. Test Case SW- 2 12](#_Toc1689223737)

[5.1.3. Test Case SW- 3 13](#_Toc635240577)

[5.1.4. Test Case SW- 4 14](#_Toc439716643)

[5.1.5. Test Case SW- 5 14](#_Toc657723402)

[5.1.6. Test Case SW- 6 15](#_Toc1939673550)

[5.1.7. Test Case SW-7 15](#_Toc984537127)

[5.2. Hardware 16](#_Toc247838773)

[5.2.1. Test Case HW-1 16](#_Toc125990918)

# Introduction

## Purpose

The GNSS Test Plan provides a structured approach to verifying the GPS display plugin's functionality within MSFS 2020. The purpose of this document is to:

* Define the scope of testing activities.
* Establish the roles and responsibilities of the testing team.
* Detail the testing goals, environment, and methodologies.

## Scope

This plan focuses on confirming the first components of the GNSS Resilience GPS display plugin, specifically:

* Ensuring the plugin integrates correctly into MSFS 2020’s instrument panel.
* Verifying the display framework is stable and ready for future functionality.

## System Overview

The GNSS Resilience GPS Display Plugin is in its first development stage. The system currently focuses on:

* Integration: Ensuring the plugin framework is embedded into the MSFS 2020 instrument panel.
* Stability Testing: Confirming that the display initializes and stays stable within the simulator.
* Preparation for Future Functionality: Laying the groundwork for displaying simulated GPS data, such as satellite status and navigation details.

## Testing Approach Overview

The testing approach focuses on confirming the GPS display plugin's framework and preparing for future functionality. The key levels of testing include:

1. Framework Stability Testing:
   * Ensures the plugin initializes and runs without errors in MSFS 2020.
2. Interface Testing:
   * Validates the visual layout and responsiveness of the GPS display within the instrument panel.
3. Preparatory Functional Testing:
   * Develops test cases for future features like satellite status and navigation data display.

Testing activities include checking the screen design, how the system runs, and how it works with MSFS 2020, as shown in Section 1.3.

## Testing Entrance Criteria

To begin system testing, the following requirements must be met:

1. The GPS display plugin framework initializes without crashes or critical errors.
2. The placeholder display appears correctly on the instrument panel with no graphical issues.
3. No critical errors occur during basic simulator interactions.
4. The GPS display plugin is integrated into MSFS 2020.
5. The testing environment meets hardware and software requirements.
6. Initial test cases for integration and stability are documented and approved.

## Document Overview

This test plan describes the approach, environment, and criteria for testing the GPS display plugin. It includes:

* Tests for initialization, placeholder display, and stability within MSFS 2020.
* Criteria for starting and stopping testing.
* A traceability matrix linking test cases to requirements.
* References to supporting documents, such as the SRS and SDD.

## References

1. GNSS System Requirements Specification (SRS), Version 2 – detailing the placeholder display and integration goals.
2. GNSS System Design Document (SDD), Version 1 – outlining system architecture and current design constraints.
3. Microsoft Flight Simulator 2020 Development Framework Documentation – providing guidelines for plugin compatibility and integration.
4. GNSS Proposal Document – defining the project scope, goals, and deliverables.

# Testing Approach

## Testing Types

The testing approach focuses on verifying the GPS display plugin’s current state while laying the groundwork for future functionality. It includes structured tests to ensure stability, visual integration, and compatibility with MSFS 2020.

### Stability Testing

* Objective: Ensure the GPS plugin initializes and runs without crashes or critical errors.
* Scope:
  + Validate that the placeholder display loads correctly in MSFS 2020.
  + Check for crashes or instability when interacting with the simulator.

### Interface Testing

* Objective: Verify that the visual and structural integration of the plugin works seamlessly within MSFS 2020.
* Scope:
  + Test the layout and alignment of the placeholder display on the instrument panel.
  + Ensure display elements are visually consistent and responsive across resolutions.

### Functionality Testing

* Objective: Confirm current and future system features, ensuring the plugin meets all requirements.
* Scope:
  + Confirm placeholder components render as expected.
  + Test future functionality, such as GPS data display, satellite status, and navigation updates.

### Preparatory Functional Testing (Future Phases)

* Objective: Develop and test foundational components to prepare for future GPS features.
* Scope:
  + Identify gaps or risks in the placeholder system that may affect future functionality.
  + Ensure readiness for navigation and satellite data simulation.

### Performance and Stress Testing (Future Phases)

* Objective: Evaluate system reliability under heavy loads.
* Scope:
  + Measure the plugin’s impact on CPU/GPU performance.
  + Test stability with rapid GPS data changes, such as frequent satellite updates or flight path shifts.

## Testing Suspension Criteria and Resumption Requirements

This section will specify the criteria used to suspend all or a part of the testing activities on the items associated with this test plan.

### Suspension Criteria

Testing will be suspended if the incidents found will not allow further testing of the system/application under-test. If testing is halted, and changes are made to the hardware, software, or database, it is up to the Testing Manager to determine whether the test plan will be re-executed, or part of the plan will be re-executed.

Testing will be suspended if:

* Plugin fails to initialize repeatedly, blocking test progression.
* Critical errors prevent loading or viewing the placeholder display.
* Environmental instability (e.g., simulator crashes) that cannot be resolved quickly.

### Resumption Requirements

Testing will resume when:

* Critical issues are resolved, and stable builds are verified.
* The simulator and plugin environment are returned to a known working state.

## Test Management Requirements

* Test Manager: Responsible for scheduling, resource allocation, approving test plans, and deciding on suspension/resumption of testing.
* Test/QA Engineer: Executes test cases, logs defects, and prepares test reports.

## Personnel Requirements

* Test Manager: Reviews and approves the test plan, manages the test schedule.
* Test/QA Engineer: Executes test procedures, records results, and logs defects.

## Testing Environment

Hardware Setup: Machine meeting MSFS 2020 minimum requirements.

Software Setup: Clean MSFS 2020 installation with the GPS plugin integrated, TypeScript debugging tools, and logging utilities.

## Testing Assumptions

* The plugin code adheres to the SRS requirements as implemented by developers.
* The test environment remains stable and unchanged during the testing window.
* Placeholders in the plugin represent future data fields that will later be populated with real navigation data.

## Testing Risks and Contingencies

* Risk: Plugin initialization fails.  
  Contingency: Perform component-level diagnostics and re-check integration steps.
* Risk: Incompatibility with MSFS updates.  
  Contingency: Test on a frozen version of MSFS until compatibility issues are resolved.
* Risk: Environment disruptions.  
  Contingency: Maintain backups and quickly revert to stable states if required.

## Cost

Testing costs are minimal at this stage, involving primarily the labor of the QA team and existing hardware and software licenses. Detailed cost estimates can be provided as needed.

## Test Plan

### Software

**Table 1: Test Plan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Test | Status[[1]](#footnote-2) | Date | Notes |
| TC-SW-001 | Plugin Initialization Test | Pending | TBD | Basic stability verification |
| TC-SW-002 | Data Display Checks | Pending | TBD | Verifies placeholder fields |
| TC-SW-003 | Refresh Rate Test | Pending | TBD | Ensures 1s update frequency |
| TC-SW-004 | Error Handling and Alerts | Pending | TBD | Tests abnormal condition handling |
| TC-SW-005 | Logging Verification | Pending | TBD | Checks GPS anomaly logging |
| TC-SW-006 | Integration & Stability Test | Pending | TBD | Examines performance impact |
| TC-SW-007 | Future-readiness Evaluation | Pending | TBD | Reviews code extensibility |

### Hardware

**Table 2: Test Plan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Test | Status[[2]](#footnote-3) | Date | Notes |
| TC-HW-001 | Hardware Button Interaction | Pending | TBD | Verifies button input and system response |

# Test Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| **Activity** | **Start Date** | **End Date** | **Responsibility** |
| Test Planning & Review | Jan 15, 2025 | Jan 25, 2025 | Test Manager |
| Environment Setup | Jan 25, 2025 | Feb 01, 2025 | Test Engineer |
| Stability & Interface Testing | Feb 01, 2025 | Feb 05, 2025 | Test Engineer |
| Functional Testing | Feb 05, 2025 | Feb 10, 2025 | Test Engineer |
| Defect Resolution & Retests | Feb 10, 2025 | Feb 15, 2025 | Test Engineer |
| Final Report & Sign-off | Feb 15, 2025 | Feb 20, 2025 | Test Manager |

# Traceability Matrix and Defect Tracking

## Traceability Matrix

**Table 1: Traceability Matrix**

|  |  |  |
| --- | --- | --- |
| Req. ID | Requirement Description | Test Case(s) |
| REQ-001 | Process satellite data & GPS signals to generate real-time navigation info | |  | | --- | | TC-001 (Initialization and Basic Data Handling) | |
| REQ-002 | |  | | --- | | Display data elements (satellite status, signal strength, coordinates, speed) | | |  | | --- | | TC-002 (Data Display Checks) | |
| REQ-003 | |  | | --- | | Update display at least once per second | | |  | | --- | | TC-003 (Refresh Rate Test) | |
| REQ-004 | |  | | --- | | Detect and respond to abnormal conditions (signal degradation, system failures) | | |  | | --- | | TC-004 (Error Handling and Alerts) | |
| REQ-005 | |  | | --- | | Log any GPS signal anomalies | | |  | | --- | | TC-005 (Logging Verification) | |
| REQ-006 | |  | | --- | | Interface seamlessly with MSFS 2020 without affecting performance | | |  | | --- | | TC-006 (Integration & Stability Test) | |
| REQ-007 | |  | | --- | | Accommodate future GNSS-related modules with minimal modifications | | TC-007 (Future-readiness Evaluation) |

## Defect Severity Definitions

|  |  |
| --- | --- |
| Critical | The defect causes a catastrophic or severe error that results in major problems and the functionality is rendered unavailable to the user. A manual procedure is impossible to implement, or high effort is required to remedy the defect. Examples of critical defects are as follows:   * System abnormally terminates * Data cannot flow through a business function/lifecycle * Data is corrupted or cannot post to the database |
| Medium | The defect that does not seriously impair system function can be categorized as a medium defect. A manual procedure requiring medium effort can be implemented to remedy the defect. Examples of medium defects are as follows:   * Form navigation is incorrect * Field labels are not consistent with global terminology |
| Low | The defect is cosmetic or has little to no impact on system functionality. A manual procedure requiring low effort can be implemented to remedy the defect. Examples of low defects are as follows:   * Repositioning of fields on screens * Text font on reports is incorrect |

# Test Cases

## Software

### Test Case SW- 1

**Test Case TC-SW-001: Plugin Initialization and Basic Data Handling**

* **Objective**: Verify that the GPS plugin loads without errors and is integrated into MSFS 2020.
* **Notes**: Ensure no crashes occur when starting MSFS 2020 with the plugin installed.
* **Test No.: TC-SW-001**
* **Current Status**: Pending
* **Test Title**: Plugin Initialization Test
* **Testing Approach**: Manual functional test within MSFS 2020 environment.

**Steps**:

1. Start MSFS 2020 with the plugin installed.
   1. **Purpose**: Confirm initialization of the plugin within the simulator.
   2. **Expected Results**: Plugin loads successfully; no error messages.
   3. **Comments**: N/A
2. Access the instrument panel where the GPS plugin is displayed.
   1. **Purpose**: Ensure the display appears as intended.
   2. **Expected Results**: Placeholder GPS interface visible on the panel.
   3. **Comments**: N/A
3. Navigate through a simple flight scenario.
   1. **Purpose**: Check if any instability or crashes occur.
   2. **Expected Results**: No crashes or performance degradation observed.
   3. **Comments**: N/A

**Concluding Remarks**: Filled in upon completion.

**Testing Team**: QA Engineer

**Date Completed**: TBD

### Test Case SW- 2

**Test Case TC-SW-002: Data Display Checks**

* **Objective**: Verify that the plugin can display placeholder data fields (satellite status, coordinates, speed) as per SRS REQ-002.
* **Notes**: Fields may be placeholders but must be visible and formatted correctly.
* **Test No.: TC-SW-002**
* **Current Status**: Pending
* **Test Title**: Placeholder Data Field Visibility
* **Testing Approach**: Manual inspection of UI elements.

**Steps**:

1. With the simulator running and the plugin loaded, observe the GPS interface fields.
   1. **Purpose**: Confirm that data fields (e.g., satellite icons) appear.
   2. **Expected Results**: Placeholder fields visible and clearly labeled.
   3. **Comments**: N/A
2. Change flight conditions (altitude, speed) to ensure fields update visually if implemented at this stage.
   1. **Purpose**: Check responsiveness of data fields.
   2. **Expected Results**: Fields remain visible and stable; placeholder text or icons remain correct.
   3. **Comments**: Actual data changes may not occur yet, just ensuring no breakage.

**Concluding Remarks**: Filled in upon completion.

**Testing Team**: QA Engineer

**Date Completed**: TBD

### Test Case SW- 3

**Test Case TC-SW-003: Refresh Rate Test**

* **Objective**: Confirm that the display refreshes at least once per second (REQ-003).
* **Test No.: TC-SW-003**
* **Current Status**: Pending
* **Test Title**: Display Refresh Rate Verification
* **Testing Approach**: Visual inspection & timing measurements.

**Steps**:

1. Start the plugin and note the time between display updates.
   1. **Purpose**: Verify the update frequency.
   2. **Expected Results**: Data/placeholder refreshed at least once per second.
   3. **Comments**: Use a stopwatch or logging output for timing.

**Concluding Remarks**: Filled in upon completion.

**Testing Team**: QA Engineer

**Date Completed**: TBD

### Test Case SW- 4

**Test Case TC-SW-004: Error Handling and Alerts**

* **Objective**: Check the plugin’s response to abnormal conditions, such as signal degradation or plugin errors (REQ-004).
* **Test No.: TC-SW-004**
* **Current Status**: Pending
* **Test Title**: Error Condition Simulation
* **Testing Approach**: Introduce mock conditions (e.g., simulate loss of signal).

**Steps**:

1. Simulate a loss of satellite visibility (if supported) or cause a known error condition.
   1. **Purpose**: Confirm system alert behavior.
   2. **Expected Results**: Plugin displays an alert message or notification.
   3. **Comments**: If simulation is not available, test this scenario when system supports it.
2. Attempt to re-initialize plugin after an error.
   1. **Purpose**: Check recovery steps.
   2. **Expected Results**: Plugin provides a retry option, recovers gracefully.
   3. **Comments**: N/A

**Concluding Remarks**: Filled in upon completion.

**Testing Team**: QA Engineer

**Date Completed**: TBD

### Test Case SW- 5

**Test Case TC-SW-005: Logging Verification**

* **Objective**: Confirm that the system logs GPS signal anomalies for post-simulation review (REQ-005).
* **Test No.: TC-SW-005**
* **Current Status**: Pending
* **Test Title**: Log File Inspection
* **Testing Approach**: Introduce a simulated anomaly and examine log files.

**Steps**:

1. Cause a known GPS anomaly (e.g., forced signal degradation).
   1. **Purpose**: Ensure an entry is recorded.
   2. **Expected Results**: A corresponding log entry detailing the anomaly.
   3. **Comments**: Confirm the correctness and clarity of log messages.
2. Review log file after the simulation.
   1. **Purpose**: Confirm log details match the anomaly.
   2. **Expected Results**: Log entry is time-stamped, descriptive, and complete.
   3. **Comments**: N/A

**Concluding Remarks**: Filled in upon completion.

**Testing Team**: QA Engineer

**Date Completed**: TBD

### Test Case SW- 6

**Test Case TC-SW-006: Integration & Stability Test**

* **Objective**: Ensure the plugin does not affect MSFS 2020’s overall performance (REQ-006).
* **Test No.: TC-SW-006**
* **Current Status**: Pending
* **Test Title**: Performance Baseline Check
* **Testing Approach**: Compare MSFS 2020 performance with and without the plugin.

**Steps**:

1. Run MSFS 2020 without the plugin and note CPU/GPU usage and framerates.
   1. **Purpose**: Establish a baseline.
   2. **Expected Results**: Baseline metrics recorded.
   3. **Comments**: N/A
2. Run MSFS 2020 with the plugin and compare performance metrics.
   1. **Purpose**: Check if plugin significantly degrades performance.
   2. **Expected Results**: Negligible or no difference in performance metrics.
   3. **Comments**: N/A

**Concluding Remarks**: Filled in upon completion.

**Testing Team**: QA Engineer

**Date Completed**: TBD

### Test Case SW-7

**Test Case TC-SW-007: Future-readiness Evaluation**

* **Objective**: Assess the codebase’s structure for accommodating future GNSS modules with minimal changes (REQ-007).
* Notes: The hardware component is intended to provide direct user input and interaction to the GPS display plugin. Each hardware control (e.g., button, valve, lever) will be tested to ensure it correctly interfaces with the software, producing the correct output or display response.
* **Test No.: TC-SW-007**
* **Current Status**: Pending
* **Test Title**: Code Structure and Extensibility Check
* **Testing Approach**: Code inspection and light architectural tests.

**Steps**:

1. Inspect the code architecture (TypeScript/HTML components).
   1. **Purpose**: Ensure modular design.
   2. **Expected Results**: Code organized into modules/classes that can be extended.
   3. **Comments**: Architecture review by developer/tester.
2. Attempt adding a mock additional data field.
   1. **Purpose**: Check effort required for new features.
   2. **Expected Results**: Simple addition without refactoring large code portions.
   3. **Comments**: N/A

**Concluding Remarks**: Filled in upon completion.

**Testing Team**: QA Engineer

**Date Completed**: TBD

## Hardware

### Test Case HW-1

**Test Case TC-HW-001: Future-readiness Evaluation**

* **Objective**: Verify the basic functionality of the hardware component and its interaction with the software component of the GNSS GPS Display Plugin. Specifically, ensure that when a physical button is pressed, it triggers the expected responses in the software display.
* **Test No.: TC-HW-001**
* **Current Status:** Pending
* **Test Title:** Hardware Button Interaction
* **Testing Approach:** Troubleshoot by operating the cockpit system’s hardware component.

**Steps:**

1. Press the button on the component.
   1. **Purpose:** Ensure basic functionality and confirm that the hardware interacts correctly with the software.
   2. **Expected Results:** When the button is pressed, the display shows the expected software response (e.g., updated data field or status change).
   3. **Comments:** As additional hardware components are integrated, similar test steps will be performed. Each button/valve/lever will be expected to result in a corresponding software action.

1.a Press the button on the component again.

* **Purpose:** Input Validation/Accuracy Testing – Ensure that the response triggered by the hardware action is valid and accurate.
* **Expected Results:** The displayed output matches the expected input criteria (e.g., proper coordinates or data).
* **Comments:** The exact expected input and output values may vary depending on the hardware component tested.

1. Press the button repeatedly in quick succession.
   1. **Purpose:** Confirm that the response time to physical hardware interactions is reasonable and prompt.
   2. **Expected Results:** The software responds instantly or within an acceptable latency threshold for each input.
   3. **Comments:** Similar tests will be conducted for each hardware control. Rapid inputs should not cause delays, missed updates, or system instability.
2. (Pending future scenarios)
   1. **Purpose:** Additional tests for new hardware components or advanced functionalities can be added here as the system evolves.
   2. **Expected Results:** (Pending)
   3. **Comments:** Further test steps will be defined once new hardware elements are integrated.

**Concluding Remarks:** Filled in by the person who completed the test.

**Testing Team:** QA Engineer

**Date Completed:** Pending

**Date Completed:** TBD

1. Unwritten, Incomplete, Pass, Fail [↑](#footnote-ref-2)
2. Unwritten, Incomplete, Pass, Fail [↑](#footnote-ref-3)