Generating Test Data From Requirements/Specifications:
Final Report

Prepared For: Rockwell Collins, Inc,
Contract Monitor: Dave Statezni

Principal Investigator: A. Jefferson Offutt
George Mason University

August 29, 1997
Update October 3, 1997
Update March 27, 1998

EXECUTIVE SUMMARY

This report presents results for the Rockwell Collins Inc. sponsored project on generating test data from requirements/specifications, which started May 19, 1997. The purpose of this project is to improve our ability to test software that needs to be highly reliable by developing formal techniques for generating test cases from formal specificalional descriptions of the software. Formal specifications represent a significant opportunity for testing because they precisely describe what functions the software is supposed to provide in a form that can be easily manipulated by automated means.

This report presents a general model for developing test inputs from state-based specifications, a derivation process for obtaining the test cases, a fully worked out example for a small system, and test cases from specifications of an industrial system. The test data generation model includes techniques for generating tests at several levels of abstraction for specifications, including the complete transition sequence level, the transition-pair level, and the detailed transition level. These techniques provide coverage criteria that are based on the specifications, and are made up of several parts, including test prefixes that contain inputs necessary to put the software into the appropriate state for the test values. The test generation process includes several steps for transforming specifications to tests.
1 INTRODUCTION

Formal specifications represent a significant opportunity for testing because they precisely describe what functions the software is supposed to provide in a form that can be easily manipulated by automated means. This research assumes that both code-based and specification-based testing are valuable, and they are complementary. It is also noted that specification-based test data generation has several advantages over code-based generation. Output checking is one of the major costs of testing, and although it is an undecidable problem, requirements/specifications can be used to partially solve it. The process of generating tests from the specifications will often help test engineers discover problems with the specifications themselves; if this step is done early, then the problems can be eliminated early, saving time and resources. Additionally, generating tests early in the development process allows testing activities to be shifted to an earlier part of the development process, allowing for more effective planning and utilization of resources.

Software functional specifications have been incorporated into testing in several ways. They have been used as a basis for test case generation, to check the output of software on test inputs, and as a basis for formalizing test specifications (as opposed to functional specifications). This research is primarily concerned with the first use, that of generating test cases from specifications. The immediate goal is to develop mechanical procedures to derive test cases from formal specifications; long term goals include automated tool support to transform formal functional specifications into effective test cases.

An abstract view of part of the test process is summarized by the diagram in Figure 1. A program $P$ (represented by a diskette), along with a set of test cases $T$, is submitted to a computer $C$, which runs $T$ on $P$ to produce some results. A primary concern for testers is how to produce $T$; the tests should be effective at finding faults in the program, adequate at providing some information about the quality of the program, and preferably satisfy some requirements or criterion for testing that is repeatable, automatable, and measurable.

A common source for tests is the program code. In code-based test generation, a testing criterion is imposed on the software to produce test requirements. For example, if the criterion of branch
testing is used, the tests are required to cover each branch in the program. An abstract view of
part of a typical test process that might be used for code-based test generation is summarized
by the diagram in Figure 2. The specification \( S \) (which can be formal or informal) is used as a
basis for writing the program \( P \), which is used to generate the tests \( T \), according to some coverage
criterion such as branch or data flow. Execution of \( T \) on \( P \) creates the \textit{actual output}, which must
be compared with the \textit{expected output}. The expected output is produced with some knowledge of
the specifications. Thus, code-based generation uses the specifications to generate the code and
decide the output of the tests.

This is in contrast to \textit{specification-based testing}, an abstract view of which is shown in Figure
3. Here the specifications are used to produce test cases, as well as to produce the program.
In this scenario, the specifications are more likely to be formalized, so the arc from \( S \) to \( P \) is
labeled “refine”, to indicate a refinement process, which is often used to create code from formal
specifications. One significance of producing tests from specifications is that the tests can be created
earlier in the development process, and be ready for execution \textbf{before} the program is finished.
Additionally, when the tests are generated, the test engineer will often find inconsistencies and
ambiguities in the specifications, allowing the specifications to be improved before the program
is written (hence the feedback arc from \( T \) back to \( S \)). The arcs from \( S \) to \( T \) and from \( S \) to
\textit{ExpectedOutput} are labeled with a “?”, because these are currently areas of active research. This
project is looking at ways to generate tests from specifications; others, such as Li et al. have been
developing techniques for creating expected output from specifications [Li96, IK95, IYZ94].

Both specification-based and code-based test generation have strengths, and both are used
in practice. Both methods have been criticized (sometimes unfairly), and both have supported
(sometimes too strongly). The Principal Investigator on this project has carried out research
involving both approaches, and this report attempts to present the strengths and weaknesses of
both approaches in a scientific, unbiased manner. While some of these are accidental differences
of the current state of the research and available technology (using the philosophical distinction
popularized by Brooks [Bro87]), some are also essential. This research takes the position that
specification-based test data generation is complementary to code-based test data generation, and
that both are necessary. Indeed most wise practical testers do this.

Because specification-based testing only considers an external view of the software, it can be said to test the products, but not the design decisions, and it may not test all of the program. By deriving tests from the specifications, it is often possible to find problems in the specifications. While this effect has not been formalized or quantified, experience has provided strong anecdotal evidence. An example is a project, called Mistix, that has been used in several classes and research projects [AO94, OI95]. It is a simplified file system program used to illustrate trees, lists, type parameterization, and inheritance. Informal specifications had been supplied to several classes before it was used as an assignment in a graduate testing class. The students applied a modification of the category-partition method [OB88, BHO89] to the specifications to produce test cases. During the exercise, they identified many inconsistencies and ambiguities in the specifications, and found several points of incompleteness. These problems helped find faults in existing implementations that had previously gone undiscovered.

Specification-based testing is also currently immature, which means there is a scarcity of formalizable criteria and automated tool support. It is this problem that this research is attempting to address. Specification-based testing has the potential to benefit from formal specifications, by using the formal specifications as input to a formalizable, automatible test generation process. Another advantage of specification-based testing is that it can support the automation of testing result analysis, by using specifications to produce expected output of test cases.

A major disadvantage of code-based generation is that it tests what was built, not what was intended. Also, code-based generation tests may not cover the entire input domain. On the other hand, code-based generation technology is very mature, and there are many formalized criteria for testing, and many tools available.

There are several things that are not known about generating tests from code and from specifications. If specification-based testing is used, it is not known how well those tests might cover the program code; likewise, we do not know how well code-based tests might cover the input domain.

The two approaches are sometimes used in combination. The most common way is to generate tests based on the specifications, and then use code-based coverage analysis to measure the quality
of the tests. For example, the tests might be measured by how many branches in the software are covered (as in Section 5.4 of this report). No results have been published concerning how effective this combination is. It is known, however, that it is difficult to construct system and subsystem level tests that cover detailed code-level requirements (such as branches). This is why code-based test generation is often thought of as useful for unit testing, when individual functions or modules are tested, and specification-based test generation is often thought of as useful for system testing, when entire working systems are tested. These are really orthogonal issues, however. Specification-based testing techniques can be and are used at the unit level. The key difference is in the questions that the two approaches attempt to answer. Specification-based testing addresses the question of “why am I testing?” whereas code-based testing addresses the question of “how much software is being covered during testing?”.

As said, this report addresses the problem of a lack of formalizable, measurable criteria for generating test cases from specifications. Specifically, a model for generating tests from requirements/specifications is presented. This report first reviews the small but growing body of work on using formal specifications as a basis for producing test cases, then presents a model for generating tests from requirements/specifications. Then a derivation process is presented, and a small example of testing a cruise control software system is given.
2 APPROACHES TO SPECIFICATION-BASED TESTING

The current research literature reports on specific tools for specific formal specification languages [BGM91, BCFG86, GMH81, Jal92, OSW86, TVK90, WGS94], manual methods for deriving tests from specifications [AA92, AO94, Ber91, DF93, Hay86], case studies on using specifications to check the output of the software on specifications [DF91, Kem85, Lay92, SC93a], and formalizations of test specifications [SC96, SC93b, BHO89, Cho86]. The term specification-based testing is used in the narrow sense of using specifications as a basis for deciding what tests to run on software. Some of these techniques are reviewed, dividing them into approaches that use model-based, algebraic, and state-based specifications.

2.1 Model-based Approaches

Model-based specification languages, such as Z and VDM, attempt to derive formal specifications of the software based on models of real-world objects. Spence and Meudec [SM] and Dick and Faivre [DF93] suggested using specifications to produce predicates, and then using predicate satisfaction techniques to generate test data. Given a set of predicates that reflect preconditions, invariants, and postconditions, test cases are generated to satisfy individual clauses. Their work was for VDM specifications, and primarily focused on state-based specifications, using finite state automata representations. Dick and Faivre discussed straightforward translation of the specifications into disjunctive normal form predicates, and presented solutions to the problem of predicate satisfaction by using prolog theorem proving techniques.

Stocks and Carrington [SC93b, SC93a] and Amla, Ammann, and Offutt [AA92, AO94] proposed using a form of domain partitioning to generate test cases. Given description of input domain, the idea is to use specifications to partition the input domain into subsets. The Amla, Ammann, and Offutt approach is based on a modification to the category-partition method for test generation [BHO89, OSW86]. Hierons [Hie97] presents algorithms that rewrite Z specifications into a form that can be used to partition the input domain. From this, states of a finite state automaton are derived, which is then used to control the test process.

Hayes [Hay86] has suggested a dynamic scheme that uses run-time verification of the program. The idea is to add code to the program to check predicates from the specifications, such as type invariants, preconditions, and input-output pairs.

2.2 Algebraic Approaches

Algebraic specification languages describe software by making formal statements, called axioms, about relationships among operations and the functions that operate on them. Gannon, McMullin and Hamlet [GMH81] used a script derivation approach. They treated the axioms as a language description and generated strings on that language to serve as test cases. Doong and Frankl [DF91] used a similar approach to test object-oriented software.

Bernot [Ber91] proposed a similar scheme, with more formalization of the process and the test cases. Bougé et al. [BCFG86] suggested a logic programming approach to generating test cases from algebraic specifications. Tsai, Volovik, and Keefe [TVK90] used a similar approach, but started with relational algebra queries.

2.3 State-based Approaches

State-based specifications describe software in terms of state transitions. Typical state-based specifications define preconditions on transitions, which are values that specific variables must have for the transition to be enabled, and triggering events, which are changes in variable values that cause the transition to be taken. For example, SCR [Hen80, AG93] calls these WHEN conditions...
and triggering events. The values the triggering events have before the transition are sometimes called *before-values*, and the values after the transition are sometimes called *after-values*. The state immediately preceding the transition is called the *pre-state*, and the *post-state* is the state after the transition.

Blackburn [BB96] used state-based functional specifications of the software, expressed in the language T-Vec, to derive disjunctive normal form constraints, similarly to Dick and Faivre’s method. These constraints are then solved to generate test cases, using special-purpose heuristic algorithms. There is a strong similarity to Blackburn’s algorithms and the algorithms used by Offutt’s test data generator [DGK+88, DO91]; the key difference being that Blackburn’s is specification-based, whereas Offutt’s constraints are code-based.

Weyuker, Goradia, and Singh [WGS94] present a method to generate test data from boolean logic specifications of software. They applied their techniques to the FAA’s Traffic Collision and Avoidance System (TCAS), and used a few mutation-style faults to measure the quality of the test cases.

### 2.4 Summary

Most of the current specification-based testing techniques use manual methods that cannot be easily generalized or automated. Goals of this research include generalizing the currently known techniques, defining measurable criteria, and developing automated tools.
3 SPECIFICATION-BASED TESTING MODEL

Section 2 discusses the notion of predicate satisfaction. Predicate satisfaction uses preconditions, invariants, and postconditions to create predicates, and then generates test cases to satisfy individual clauses within the predicates. This is closely related to previous code-based automatic test generation research [DO91]. The model presented here extends the promising ideas of predicate satisfaction in several ways.

Instead of just covering the pre and postconditions, it is important to use the tests to relate the preconditions to the postconditions. Tests should also be created to find faults, as well as to cover the input domain. This report presents examples using Software Cost Reduction specifications (SCR) [Hen80, AG93] and CoRE [FBWK92].

In this model, tests are generated as multi-part, multi-step, multi-level artifacts. The multi-part aspect means that a test case is composed of several components. Input values are the values for the test case; these are the values needed to directly satisfy the test requirements. The other components supply supporting values, including expected outputs, inputs necessary to get to the appropriate pre-state, and inputs necessary to observe the effect of the test case. The multi-step aspect means that tests are generated in several steps from the functional specifications by a refinement process. The functional specifications are refined into test specifications, which are then refined into test scripts. The multi-level aspect means that tests are generated to test the software at several levels of abstraction.

The multiple parts of the test case are based on research in test case specifications [BHO89, SC93b]. The category-partition method includes a test specification language called TSL [BHO89], which was designed for command-line interface software. A test case in TSL is a command or software function and values for its parameters and relevant environment variables. A test specification in TSL consists of the operations necessary to create the environmental conditions (called the SETUP portion), the test case itself, whatever command is necessary to observe the effect of the operation (VERIFY in TSL), and any exit command (CLEANUP in TSL). Test specifications written in TSL can be used to automatically generate executable test scripts that are ready for input to the software.

In this state-based approach, the test case operation of TSL is replaced by Test case values, which are derived directly from a triggering event and the preconditions for the transition. The setup operation is called a Prefix, and includes all inputs necessary to reach the pre-state and to give the triggering event variable its before-value. Any inputs that are necessary to show the results are Verify operations, and Exit commands depend on the system being tested. Expected outputs are created from the after-values of the triggering events and any postconditions that are associated with the transition.

The model currently defines test cases at four levels: (1) the transition coverage level, (2) the full predicate coverage level, (3) the transition-pair coverage level, and (4) the complete sequence level. These are defined in the next four subsections. To apply these, a state-based requirement/specification is viewed as a directed graph, called the specification graph. Each node represents a state (or mode) in the requirement/specification, and edges represent possible transitions.

It is possible to apply all levels, or to choose a level based on a cost/benefit tradeoff. The first two are related; the transition coverage level requires many fewer test cases than the full predicate coverage level, but if the full predicate coverage level is used, the tests will also satisfy the transition coverage level (full predicate coverage subsumes transition coverage). Thus only one of these two should be used. The latter two levels are meant to be independent; transition-pair coverage is intended to check the interfaces among states, and complete sequence testing is intended to check the software by executing the software through complete execution paths. As it happens, transition-pair coverage subsumes transition coverage, but they are designed to test the software in very different ways.
3.1 Transition Coverage Level

This level is analogous to the branch coverage criterion in structural testing. The requirement is that each transition in the specification graph is taken at least once. This is another way of requiring that each precondition in the specification is satisfied at least once.

| Transition coverage: Each transition in the specification graph is taken at least once. |

3.2 Full Predicate Coverage Level

One question during testing is whether the predicates in the specifications are formulated correctly. Small inaccuracies in the specification predicates can lead to major problems in the software. The full predicate coverage level takes the philosophy that to test the software, we should at least provide inputs to test each clause in each predicate. This level requires that each clause in each predicate on each transition is tested independently, thus attempting to address the question of whether each clause is necessary and is formulated correctly. Following the definitions in DO178B [SC-92], the Boolean operators are AND, OR, and NOT, and clause and predicate are defined as follows (DO178B uses the terms “condition” and “decision”):

- A clause is a Boolean expression that contains no Boolean operators. For example, relational expressions and Boolean variables are clauses.
- A predicate is a Boolean expression that is composed of clauses and zero or more Boolean operators. A predicate without a Boolean operator is also a clause. If a clause appears more than once in a predicate, each occurrence is a distinct clause.

The concept of full predicate coverage is based on the structural testing criterion of modified condition/decision coverage (MC/DC) [CM94], which requires that every decision and every condition within the decision has taken every outcome at least once, and every condition has been shown to independently affect its decision. The full predicate coverage level is defined as follows:

| Full predicate coverage: Each clause in turn takes the values of True and False while all other clauses in the predicate have values such that the value of the predicate will always be the same as clause being tested. |

This definition ensures that each clause is tested without being influenced by the other clauses. Note that if full predicate coverage is achieved, transition coverage will also be achieved. To satisfy the requirement that the test clause controls the value of the predicate, other clauses must be either True or False. If the predicate is \((X \land Y)\), and the test clause is \(X\), then \(Y\) must be \text{True}. Likewise, if the predicate is \((X \lor Y)\), \(Y\) must be \text{False}.

The simplest way to satisfy full predicate is to use an expression parse tree. An expression parse tree is a binary tree that has binary and unary operators for internal nodes, and variables or constants at leaf nodes. The relevant binary operators are \text{and} \((\land)\), \text{or} \((\lor)\), and the relational operators \(\{ >, <, \leq, \geq, =, \neq \}\); the relevant unary operator is \text{not}. For example, the expression parse tree for \((A \lor B) \land C\) is:
Given a parse tree, full predicate coverage is satisfied by walking the tree. First, a test clause is chosen. Then the parse tree is walked from the test clause up to the root, then from the root down to each clause. If its parent is \( \lor \), its sibling must have the value of \( \text{False} \), if its parent is \( \land \), its sibling must have the value of \( \text{True} \). If a node is the inverse operator \( \text{not} \), the parent node is given the inverse value of the child node. This is repeated for each node between the test clause and the root.

Once the root is reached, values can be propagated back down using a simple tree walk. If a \( \land \) node has the value of \( \text{True} \), then both children must have the value \( \text{True} \); if a \( \land \) node has the value of \( \text{False} \), then either child must have the value \( \text{False} \) (which one is arbitrary). If a \( \lor \) node has the value of \( \text{False} \), then both children must have the value \( \text{False} \); if a \( \lor \) node has the value of \( \text{True} \), then either child must have the value \( \text{True} \) (which one is arbitrary). If a node is the inverse operator \( \text{not} \), the parent node is given the inverse value of the child node.

Figure 4 illustrates the process for the expression above, showing both \( B \) and \( C \) as test clauses. In the top sequence, \( B \) is the test clause (shown with a dashed box). In tree 2, its sibling, \( A \), is assigned the value \( \text{False} \), and in tree 3, \( C \) is assigned the value \( \text{True} \). In the bottom sequence, \( C \) is the test clause. In tree 2, \( C \)'s sibling is a \( \land \) node, and is assigned the value \( \text{True} \). In tree 3, \( A \) is assigned the value \( \text{True} \). Note that in tree 3, either \( A \) or \( B \) could be given the \( \text{True} \) value; the choice is arbitrary.

Although this may seem complicated, it is easy to automate and relatively straightforward to apply by hand. It has also been our experience that most specification predicates tend to be fairly small and simple in form.
These test cases sample from both valid and invalid transitions, with only one transition being valid at a time. In addition, the test engineer may choose semantically meaningful combinations of conditions. Testing with invalid inputs can help find faults in the implementation as well as the formulation of the specifications. Of course, this brings up a philosophical question about responsibility. Many developers believe that if a software component has well-stated preconditions, it is the responsibility of the user to ensure that the preconditions are met. This can be taken to imply that the component does not need to be tested with inputs that violate the preconditions (as in the design-by-contract approach [MM92]). Without taking a side on this issue, the technique described here provides a mechanism for developing invalid inputs; they can be used or discarded as the tester sees fit.

As a concrete example, consider the formula whose parse tree was given above, \((A \lor B) \land C\). The following partial truth table provides the values for the test clauses:

\[
\begin{array}{ccc}
(A \lor B) \land C \\
1 & T & T \\
2 & F & F \\
3 & T & T \\
4 & F & F \\
5 & T & T \\
6 & T & F \\
\end{array}
\]

To ensure the requirement that the test clause must control the final result, the partial truth table must be filled out as follows (for the last two entries, either \(A\) or \(B\) could have been \(\text{True}\), both were assigned the value \(\text{True}\)):

\[
\begin{array}{ccc}
(A \lor B) \land C \\
1 & T & F & T \\
2 & F & F & F \\
3 & F & T & T \\
4 & F & F & F \\
5 & T & T & T \\
6 & T & F & F \\
\end{array}
\]

Some specification languages, such as SCR and CoRE, treat triggering event variables differently from other variables in transition predicates. When this is the case, the clause that corresponds to the triggering event should be given a different value, but should remain the triggering event. If it is no longer a triggering event, it is equivalent to not executing a test case. Moreover, a triggering event actually specifies two values, a before-value and an after-value. To fully test predicates with triggering events, both before- and after-values should be tested. This is done by assuming two versions of the triggering event, \(A\) and \(A'\), where \(A\) represents the before-value of \(A\) and \(A'\) represents its after-value.

### 3.3 Transition-Pair Coverage Level

The previous testing levels test transitions independently, but do not test sequences of state transitions. This level requires that pairs of transitions be taken.
**Transition-pair coverage level**: For each state $S$, form test requirements such that for each incoming transition and each outgoing transition, both transitions must be taken sequentially.

Consider the following state:

![State Diagram]

To test the state $S$ at the transition-pair level, six tests are required: (1) from 1 to i, (2) 2 to i, (3) 1 to ii, (4) 2 to ii, (5) 1 to iii, and (6) 2 to iii. These tests require inputs that satisfy the predicates: $P_1:P_i$, $P_1:P_{ii}$, $P_1:P_{iii}$, $P_2:P_i$, $P_2:P_{ii}$, and $P_2:P_{iii}$.

### 3.4 Complete Sequence Level

It seems very unlikely that any successful test method could be based on purely mechanical methods; at some point the experience and knowledge of the test engineer must be used. Particularly at the system level, effective testing probably requires detailed domain knowledge. A complete sequence is a sequence of state transitions that form a complete practical use of the system. This use of the term is similar to that of “use cases”. In most realistic applications, the number of possible sequences is too large to choose all complete sequences. In many cases, the number of complete sequences is infinite.

**Complete sequence level**: The test engineer must define meaningful sequences of transitions on the specification graph by choosing sequences of states that should be entered.

Which sequences to choose is something that can only be determined by the test engineer with the use of domain knowledge and experience. This is the least automatable level of testing.

### 3.5 Summary

To generate tests according to this methodology, tests must be generated at the following four levels:

1. **Transition Coverage Level**
   - Definition: Each transition in the specification graph is taken at least once.
   - Requirements: Predicates on the edges must evaluate to True.
   - Specifications:
     - Prefix: Inputs to get to the pre-state immediately preceding the edge.
     - Test case values: Assignments to variables to satisfy the preconditions and a new value for the triggering event variable.
     - Verify: Input to the software to show the post-state; depends on the software.
– Exit: Input to the software to stop execution; depends on the software.
– Expected outputs: Post-state from the requirements.

• Script: A sequence of inputs to the software; the format depends on the software.

2. Full Predicate Coverage Level

• Definition: Each clause in turn takes the values of \texttt{True} and \texttt{False} while all other clauses in the predicate have values such that the value of the predicate will always be the same as clause being tested.

• Requirements: Certain rows from the truth tables of the predicates must be chosen.

• Specifications:
  – Prefix: Inputs to get to the pre-state immediately preceding the edge.
  – Test case values: Assignments to variables to satisfy the preconditions and a new value for the triggering event variable.
  – Verify: Inputs to the software to show the post-state; depends on the software.
  – Exit: Input to the software to stop execution; depends on the software.
  – Expected outputs: Post-state from the requirements.

• Script: A sequence of inputs to the software; the format depends on the software.

3. Transition-Pair Coverage Level

• Definition: For each state \( S \), form test requirements such that for each incoming transition and each outgoing transition, both transitions must be taken in sequence.

• Requirements: Predicates on two edges of the specification graph must evaluate to \texttt{True}.

• Specifications:
  – Prefix: Inputs to get to the pre-state immediately preceding the edge.
  – Test case values: Assignments to variables to satisfy the preconditions and a new value for the triggering event variable.
  – Verify: Inputs to the software to show the post-state; depends on the software.
  – Exit: Input to the software to stop execution; depends on the software.
  – Expected outputs: Post-state from the requirements.

• Script: A sequence of inputs to the software; the format depends on the software.

4. Complete Sequence Level

• Definition: The test engineer must define \texttt{meaningful} sequences of transitions on the specification graph by choosing sequences of states that should be entered.

• Requirements: Lists of states.

• Specifications:
  – Setup: Should be empty
  – Test case value: Value assignments necessary to take every transition on the sequence path.
  – Verify: Inputs to the software to show the post-state; depends on the software.
  – Exit: Input to the software to stop execution; depends on the software.
  – Expected outputs: Sequence of states.

• Script: A sequence of inputs to the software; the format depends on the software.
4 DERIVATION PROCESS

This section presents a process that can be used to derive test cases. The process steps for all four levels of testing are presented together, as there is a fair amount of overlap. If not all four levels are used, some of these steps should be skipped. The steps are presented as being purely manual; in the future schemes for automating as many of the steps as possible will be developed.

The general process is shown in Figure 5; this merely reflects the multi-step aspect of our test generation process that was presented in Section 3.

![Diagram of test case generation process]

1. Develop transition conditions. The first step is to develop transition conditions, which are predicates that define under what conditions each transition will be taken. With some specification languages (e.g., SCR and CoRE), the transition conditions are encoded directly into the specifications. With other languages, the conditions may have to be derived.

2. Develop specification graph. The specification graph was described in Section 3. It can be directly derived from the specification table, and edges annotated with the conditions derived in step 1.

   This is the point at which the process separates for the four testing levels.

3. Develop transition coverage test requirements.

   (a) Derive transition predicates. The conditions from step 1 are listed one at a time to form test requirements.

4. Develop full predicate test requirements.

   (a) Construct truth tables for all predicates in the specification graph. The predicate coverage tests can be based on an expression tree or directly on the predicates. If all the logical connectors are the same (all ANDs or all ORs), it is a simple matter to modify the values for the clauses in the predicates directly. If ANDs and ORs are mixed freely, however, it is less error-prone to construct the expression tree. Most specification languages differentiate between trigger events and preconditions; in this case, the trigger events must be marked specially so that the test engineer remembers to put that input after the precondition inputs.
5. Develop transition-pair test requirements.

(a) Identify all pairs of transitions. Transition-pair tests are ordered pairs of condition values, each representing an input to the state and an output from the state. These are formed by enumerating all the input transitions \( M \), all the output transitions \( N \), then creating \( M \times N \) pairs of transitions.

(b) Construct predicate pairs. These pairs of transitions are then replaced by the predicates from the specification graph.

6. Develop complete sequence test requirements.

(a) Identify complete lists of states. The complete sequence tests are created by the tester. This is done by choosing sequences of states from the specification graph to enter.

(b) Construct sequence of predicates. The sequences of states are transformed into sequences of conditions that will cause those states to be entered.

At this point, test requirements for the four levels will be in a uniform format, as truth assignments for predicates.

7. Construct test specifications. For each unique test requirement, generate prefix values, test case values, verify conditions, exit conditions, and expected outputs. Note that there may be a fair amount of overlap among the test requirements, thus the "unique" restriction. Generating the actual values may involve solving some algebraic equations. For example, if a condition is \( A > B \), values for \( A \) and \( B \) must be chosen to give the predicate the appropriate value. It is also at this point that some "invalid" tests might be discovered. For example, it may be impossible or meaningless to pair all incoming and outgoing transitions for each state. In this case, certain test specifications will be discarded.

8. Construct test scripts. Each test specification is used to construct one test script. The actual scripts must reflect the input syntax of the program, so knowledge of the input syntax of the program is required for this step. (Note that this is the only step that requires any knowledge of the implementation, all preceding steps depend solely on the functional specifications.)

4.1 Automation Notes

It is possible to automate almost all of this derivation process. If a machine-readable form of the specification table is available, the transition conditions can be read directly from the table. The specification graph can then be automatically created from the states and transition conditions. Test requirements take the form of partial truth tables defined on transition predicates, state transition predicates, and pairs of state transition predicates. Given a formal functional specification, most if not all of these test requirements can be generated automatically. The prefix of a test case includes inputs necessary to put the system into a particular pre-state. Given the specification graph, many of these prefixes can be generated automatically. One open question is whether this problem is generally solvable (unlike the related readability problem in general software, which is generally unsolvable), and how to solve or partially solve the problem. It is also possible to automatically refine test specifications into test scripts. Finally, algorithms for automatically generating test scripts can be developed, although the input syntax of the program will be needed. The final step, generating complete sequence tests, cannot be fully automated. But an appropriate interface could present the specification graph, and allow the tester to choose sequences of states by pointing and clicking on the screen. Each time a state is chosen, the transition from the previous state could be automatically translated into values and saved as part of the test case. This would allow the tester's job to become the purely intellectual exercise of choosing sequences of states to be entered.
5 CRUISE CONTROL EXAMPLE

This section presents an example of applying the test data generation model to a specification for an automobile cruise control system. Cruise control is a common example in the literature [Atti94, Jin96]. Table 1 shows the specifications for the system (note that it does not model the throttle). It has four states: OFF (the initial state), INACTIVE, CRUISE, and OVERRIDE. The system’s environmental conditions indicate whether the automobile’s ignition is on (Ignited), the engine is running (Running), the automobile is going too fast to be controlled (TooFast), the brake pedal is being pressed (Brake), and whether the cruise control level is set to Activate, Deactivate, or Resume.

<table>
<thead>
<tr>
<th>Previous Mode</th>
<th>Ignited</th>
<th>Running</th>
<th>Toofast</th>
<th>Brake</th>
<th>Activate</th>
<th>Deactivate</th>
<th>Resume</th>
<th>New Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off</td>
<td>@T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Off</td>
</tr>
<tr>
<td>Inactive</td>
<td>@F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Inactive</td>
</tr>
<tr>
<td>Cruise</td>
<td>@F</td>
<td>-</td>
<td>-</td>
<td>@T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Cruise</td>
</tr>
<tr>
<td></td>
<td>@F</td>
<td>@F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Inactive</td>
</tr>
<tr>
<td></td>
<td>@T</td>
<td>@T</td>
<td>@F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Override</td>
</tr>
<tr>
<td>Override</td>
<td>@F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Off</td>
</tr>
<tr>
<td></td>
<td>@F</td>
<td>-</td>
<td>-</td>
<td>@T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Inactive</td>
</tr>
<tr>
<td></td>
<td>@T</td>
<td>-</td>
<td>-</td>
<td>@F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Cruise</td>
</tr>
</tbody>
</table>

Table 1: SCR Specifications for the Cruise Control System

Each row in the table specifies a conditioned event that activates a transition from the mode on the left to the mode on the right. A table entry of @T or @F under a column header C represents a triggering event @T(C) or @F(C). This means that the value of C must change for the transition to be taken. A table entry of t or f represents a WHEN condition. WHEN[C] means the transition can only be taken if C is true, and WHEN[¬C] means it can only be taken if C is false. If the value of an environmental condition C does not affect a conditioned event, the table entry is marked with a hyphen “-” (don’t care conditions).

Table 2 shows the transitions of the specification in predicate form, numbered $P_1$ through $P_{12}$. Figure 6 shows the specification graph, with the edges labeled with the condition numbers.

5.1 Full Predicate Coverage Level

There are nine transitions in the cruise control specifications, and twelve disjunctive predicates (Table 1 shows each disjunctive predicate on a separate line). For convenience, the technique is applied by considering each predicate specification separately. As stated in Section 3.2, both the before- and after-values of the triggering event should be tested. For SCR, this is handled by treating @ as an operator and expanding it algebraically. The relevant expansions are:

- $@T(X) \equiv X \land X'$
- $@T(X \land Y) \equiv \neg(X \land Y) \land (X' \land Y') \equiv (\neg X \lor \neg Y) \land X' \land Y'$
- $@T(X \lor Y) \equiv \neg(X \lor Y) \land (X' \land Y') \equiv \neg X \land \neg Y \land X' \land Y'$

Table 3 repeats Table 2, but with the trigger events expanded appropriately.

The test case requirements for the full predicate coverage level are below with the environmental variables shown as I (Ignited) R (Running), T (TooFast), B (Brake), A (Activate), D (Deactivate),
\[ P_1 \text{ OFF} @T_{Ignited} \quad \text{INACTIVE} \]
\[ P_2 \text{ INACTIVE} @F_{Ignited} \quad \text{OFF} \]
\[ P_3 \text{ INACTIVE} @T_{Activate} \land T_{Ignited} \land T_{Running} \land \neg T_{Brake} \quad \text{CRUISE} \]
\[ P_4 \text{ CRUISE} @F_{Ignited} \quad \text{OFF} \]
\[ P_5 \text{ CRUISE} @F_{Running} \land T_{Ignited} \quad \text{INACTIVE} \]
\[ P_6 \text{ CRUISE} @T_{TooFast} \land T_{Ignited} \quad \text{INACTIVE} \]
\[ P_7 \text{ CRUISE} @T_{Brake} \land T_{Ignited} \land T_{Running} \land \neg T_{TooFast} \quad \text{OVERRIDE} \]
\[ P_8 \text{ CRUISE} @T_{Deactivate} \land T_{Ignited} \land T_{Running} \land \neg T_{TooFast} \quad \text{OVERRIDE} \]
\[ P_9 \text{ OVERRIDE} @F_{Ignited} \quad \text{OFF} \]
\[ P_{10} \text{ OVERRIDE} @F_{Running} \land T_{Ignited} \quad \text{INACTIVE} \]
\[ P_{11} \text{ OVERRIDE} @T_{Activate} \land T_{Ignited} \land T_{Running} \land \neg T_{Brake} \quad \text{CRUISE} \]
\[ P_{12} \text{ OVERRIDE} @T_{Resume} \land T_{Ignited} \land T_{Running} \land \neg T_{Brake} \quad \text{CRUISE} \]

Table 2: Cruise Control Specification Predicates

Figure 6: Specification Graph for Cruise Control
and $S$ (Resume). The variable values are taken from the predicates, and are shown as T, F, t, f, and -. A T or F means the clause is triggering, and the table contains a before-and after-value. The values for the test case are the new value for the triggering clause (T or F), and the t and f values from the WHEN conditions. The expected output for the test specification is derived from the triggering event, the post-state, and any terms or variables that are defined as a result of the transition.

The first two transitions have only one clause, so the only test cases are based on the triggering event. The third transition, $P_3$, has four clauses:

@T Activate ∧ Ignited ∧ Running ∧ ¬Brake

and its expanded version is:

¬Activate ∧ Ignited ∧ Running ∧ ¬Brake ∧ Activate’

Its test case requirements are:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>Activate</th>
<th>Ignited</th>
<th>Running</th>
<th>Brake</th>
<th>Activate’</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>INACTIVE</td>
<td>F</td>
<td>t</td>
<td>t</td>
<td>f</td>
<td>T</td>
<td>CRUISE</td>
</tr>
<tr>
<td>INACTIVE</td>
<td>T</td>
<td>t</td>
<td>t</td>
<td>f</td>
<td>T</td>
<td>INACTIVE</td>
</tr>
<tr>
<td>INACTIVE</td>
<td>F</td>
<td>f</td>
<td>t</td>
<td>f</td>
<td>T</td>
<td>OFF</td>
</tr>
<tr>
<td>INACTIVE</td>
<td>F</td>
<td>t</td>
<td>f</td>
<td>f</td>
<td>T</td>
<td>INACTIVE</td>
</tr>
<tr>
<td>INACTIVE</td>
<td>F</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>T</td>
<td>INACTIVE</td>
</tr>
<tr>
<td>INACTIVE</td>
<td>F</td>
<td>t</td>
<td>t</td>
<td>f</td>
<td>F</td>
<td>INACTIVE</td>
</tr>
</tbody>
</table>

The first row is the predicate as it appears in the specification; every clause is True. The subsequent rows make each clause false in turn. Because there are no ∨ operators, the full predicate coverage criterion is satisfied by holding all other clauses True.

Below are the requirements for all the predicates in the cruise control program. There are 54 test cases for the 12 predicates.
<table>
<thead>
<tr>
<th>Pre State</th>
<th>Variable Values</th>
<th>Triggering Event</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>OFF</td>
<td>Ignited = True</td>
<td>INACTIVE</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>Ignited = True</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>Ignited = False</td>
<td>OFF</td>
</tr>
<tr>
<td>P2</td>
<td>INACTIVE</td>
<td>Ignited = False</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>INACTIVE</td>
<td>Ignited = False</td>
<td>INACTIVE</td>
</tr>
<tr>
<td></td>
<td>INACTIVE</td>
<td>Ignited = True</td>
<td>INACTIVE</td>
</tr>
<tr>
<td>P3</td>
<td>INACTIVE</td>
<td>Activate = True</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>INACTIVE</td>
<td>Activate = True</td>
<td>INACTIVE</td>
</tr>
<tr>
<td></td>
<td>INACTIVE</td>
<td>Activate = True</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>INACTIVE</td>
<td>Activate = True</td>
<td>INACTIVE</td>
</tr>
<tr>
<td></td>
<td>INACTIVE</td>
<td>Activate = False</td>
<td>INACTIVE</td>
</tr>
<tr>
<td>P4</td>
<td>CRUISE</td>
<td>Ignited = False</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Ignited = False</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Ignited = True</td>
<td>CRUISE</td>
</tr>
<tr>
<td>P5</td>
<td>CRUISE</td>
<td>Running = False</td>
<td>INACTIVE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Running = False</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Running = False</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Running = True</td>
<td>CRUISE</td>
</tr>
<tr>
<td>P6</td>
<td>CRUISE</td>
<td>TooFast = True</td>
<td>INACTIVE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>TooFast = True</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>TooFast = True</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>TooFast = False</td>
<td>CRUISE</td>
</tr>
<tr>
<td>P7</td>
<td>CRUISE</td>
<td>Brake = True</td>
<td>OVERRIDE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Brake = True</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Brake = True</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Brake = True</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Brake = False</td>
<td>CRUISE</td>
</tr>
<tr>
<td>P8</td>
<td>CRUISE</td>
<td>Deactivate = True</td>
<td>OVERRIDE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Deactivate = True</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Deactivate = True</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Deactivate = True</td>
<td>CRUISE</td>
</tr>
<tr>
<td></td>
<td>CRUISE</td>
<td>Deactivate = False</td>
<td>CRUISE</td>
</tr>
<tr>
<td>P9</td>
<td>OVERRIDE</td>
<td>Ignited = False</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>OVERRIDE</td>
<td>Ignited = False</td>
<td>OVERRIDE</td>
</tr>
<tr>
<td></td>
<td>OVERRIDE</td>
<td>Ignited = True</td>
<td>OVERRIDE</td>
</tr>
</tbody>
</table>
5.1.1 Test specifications

The actual test specifications and test scripts are mechanically derived from the above test requirements, and are too numerous to list. The predicate P3 is chosen as an illustrative example. P3 has six full predicate level tests. For the first test case for P3, the test case must reach the INACTIVE state; this forms the Prefix. The Test case values set the before-value for the triggering event, and the WHEN condition variables of Inactive, Running, and Brake, and then sets Activate to be True as the triggering event. The Verify and Exit parts of the specifications are not shown, as they depend on the software. The software can safely be assumed to automatically print the (post) current state, and to not require an exit.

1. Test specification P3-1:

Prefix: \( \text{Ignited} = \text{True} \) – Reach INACTIVE state
Test case value: \( \text{Activate} = \text{False} \) – Trigger before-value
\( \text{Running} = \text{True} \) – Condition variable
\( \text{Brake} = \text{False} \) – Condition variable
\( \text{Activate} = \text{True} \) – Triggering event

Expected outputs: CRUISE

2. Test specification P3-2:

Prefix: \( \text{Ignited} = \text{True} \) – Reach INACTIVE state
Test case value: \( \text{Activate} = \text{False} \) – Trigger before-value
\( \text{Ignited} = \text{False} \) – Condition variable
\( \text{Running} = \text{True} \) – Condition variable
\( \text{Brake} = \text{False} \) – Condition variable
\( \text{Activate} = \text{True} \) – Triggering event

Expected outputs: INACTIVE
3. Test specification P3-3:

Prefix: \( \text{Ignited} = \text{True} \) – Reach INACTIVE state
Test case value: \( \text{Activate} = \text{False} \) – Trigger before-value
\( \text{Running} = \text{False} \) – Condition variable
\( \text{Brake} = \text{False} \) – Condition variable
\( \text{Activate} = \text{True} \) – Triggering event

Expected outputs: OFF

4. Test specification P3-4:

Prefix: \( \text{Ignited} = \text{True} \) – Reach INACTIVE state
Test case value: \( \text{Activate} = \text{False} \) – Trigger before-value
\( \text{Running} = \text{True} \) – Condition variable
\( \text{Brake} = \text{True} \) – Condition variable
\( \text{Activate} = \text{True} \) – Triggering event

Expected outputs: INACTIVE

5. Test specification P3-5:

Prefix: \( \text{Ignited} = \text{True} \) – Reach INACTIVE state
Test case value: \( \text{Activate} = \text{True} \) – Trigger before-value
\( \text{Running} = \text{True} \) – Condition variable
\( \text{Brake} = \text{False} \) – Condition variable
\( \text{Activate} = \text{True} \) – Triggering event

Expected outputs: INACTIVE

6. Test specification P3-6:

Prefix: \( \text{Ignited} = \text{True} \) – Reach INACTIVE state
Test case value: \( \text{Activate} = \text{False} \) – Trigger before-value
\( \text{Running} = \text{True} \) – Condition variable
\( \text{Brake} = \text{False} \) – Condition variable
\( \text{Activate} = \text{False} \) – Triggering event

Expected outputs: INACTIVE

There are several interesting points to note about these test specifications. First, it should be clear that there is some redundancy; some of the condition variables will not need to be explicitly set, as they will already have the appropriate values. While this is true, the analysis necessary to decide what values do and do not need to be set probably outweighs the small savings that could result from eliminating a few variable assignments. It is probable, however, that this could be done automatically. Jin [Jin96] provided algorithms for deriving invariants on modes; these could be used to directly eliminate unneeded variable assignments. Her method used a static analysis. A dynamic analysis that uses the information in the test specification could be used to potentially...
eliminate more variable assignments. Another interesting point is the derivation of the prefix part of the test specification. Reaching the pre-state is essentially a reachability problem. Given a control flow graph of a program, it is an undecidable problem to find a test case that reaches a particular statement. Although no theoretical analysis has been done as yet, it seems likely that the deterministic nature of state-based systems means that this problem is solvable for specification graphs derived from state-based systems.

Test scripts are simple rewrites of test specifications with modifications made for the input requirements of the program being tested. The test script for the first test specification above is:

Ignited = True
Activate = False
Running = True
Brake = False
Activate = True

5.2 Transition-Pair Coverage Level

At the transition-pair level, each state is considered separately. Each input transition into the state is matched with each transition out of the state, and the combination is used to create test requirements, which are ordered pairs of predicates. The ordered pairs are turned into ordered pairs of inputs to form test specifications.

Following are the test requirements for the four states. The pairs for the OFF state are:

1. P2:P1
2. P4:P1
3. P9:P1

The pairs for the INACTIVE state are:

1. P1:P2
2. P1:P3
3. P10:P2
4. P10:P3
5. (P5 OR P6):P2
6. (P5 OR P6):P3

The pairs for the CRUISE state are:

1. P3:P4
2. P3:(P5 OR P6)
3. P3:(P7 OR P8)
4. (P11 OR P12):P4
5. (P11 OR P12):(P5 OR P6)
6. (P11 OR P12):(P7 OR P8)

The pairs for the OVERRIDE state are:

1. (P7 OR P8):P9
2. (P7 OR P8):P10
3. (P7 OR P8):(P11 OR P12)
These ordered pairs are transformed into predicates from Table 2. The “OR” entries result from the transitions that have two conditions; either condition could be satisfied to take that transition. Rather than list before- and after-values for the triggering events in this table, only the after-values are shown; the before-values are assumed to be the inverse.

<table>
<thead>
<tr>
<th>I R T B A D S</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OFF:</strong></td>
</tr>
<tr>
<td>1. INACTIVE</td>
</tr>
<tr>
<td>OFF</td>
</tr>
<tr>
<td>2. CRUISE</td>
</tr>
<tr>
<td>3. OVERRIDE</td>
</tr>
<tr>
<td><strong>OFF</strong></td>
</tr>
<tr>
<td><strong>INA CTIVE:</strong></td>
</tr>
<tr>
<td>1. OFF</td>
</tr>
<tr>
<td>2. OFF</td>
</tr>
<tr>
<td>3. OVERRIDE</td>
</tr>
<tr>
<td>4. OVERRIDE</td>
</tr>
<tr>
<td>5. CRUISE</td>
</tr>
<tr>
<td><strong>CRUISE:</strong></td>
</tr>
<tr>
<td>1. INACTIVE</td>
</tr>
<tr>
<td>2. INACTIVE</td>
</tr>
<tr>
<td>3. CRUISE</td>
</tr>
<tr>
<td>4. CRUISE</td>
</tr>
<tr>
<td>5. OVERRIDE</td>
</tr>
<tr>
<td>6. OVERRIDE</td>
</tr>
</tbody>
</table>
OR
OVERRIDE  t t - f   T  CRUISE
CRUISE  t t f T - - -  OVERRIDE
OR
CRUISE  t t f - - T  OVERRIDE

OVERRIDE:  1. CRUISE  t t f T - - -  OVERRIDE
OR
CRUISE  t t f - - T  OVERRIDE
OVERRIDE  F - - - - -  OFF
2. CRUISE  t t f T - - -  OVERRIDE
OR
CRUISE  t t f - - T  OVERRIDE
OVERRIDE  t F - - - - -  INACTIVE
3. CRUISE  t t f T - - -  OVERRIDE
OR
CRUISE  t t f - - T  OVERRIDE
OVERRIDE  t t - f T - -  CRUISE
OR
OVERRIDE  t t - f - T  CRUISE

5.2.1 Test specifications

The actual test specifications and test scripts are mechanically derived from the above test requirements, and are too numerous to list. The requirements for the OFF state are chosen as an illustrative example. OFF has four transition-pair coverage level tests. For the first test case for OFF, the test case must reach the INACTIVE state; this forms the Prefix. Then the test case must pass through transitions P1 and P2.

1. Test specification OFF-1:

Prefix:  \( Ignited = True \) – Reach INACTIVE state
Test case values:  \( Ignited = False \) – P2 Triggering event
\( Ignited = True \) – P1 Triggering event
Expected outputs:  INACTIVE

2. Test specification OFF-2:

Prefix:  \( Ignited = True \) – Reach INACTIVE state
\( Ignited = True \) – P3 Condition variable
\( Running = True \) – P3 Condition variable
\( Brake = False \) – P3 Condition variable
\( Activate = True \) – Reach CRUISE state
Test case values:  \( Ignited = False \) – P4 Triggering event
\( Ignited = True \) – P1 Triggering event
Expected outputs:  INACTIVE
3. Test specification OFF-3:

Prefix:  
\[\begin{align*}
\text{Ignited} &= \text{True} \quad \text{Reach INACTIVE state} \\
\text{Ignited} &= \text{True} \quad \text{P3 Condition variable} \\
\text{Running} &= \text{True} \quad \text{P3 Condition variable} \\
\text{Brake} &= \text{False} \quad \text{P3 Condition variable} \\
\text{Activate} &= \text{True} \quad \text{Reach CRUISE state} \\
\text{Ignited} &= \text{True} \quad \text{P7 Condition variable} \\
\text{Running} &= \text{True} \quad \text{P7 Condition variable} \\
\text{TooFast} &= \text{False} \quad \text{P7 Condition variable} \\
\text{Brake} &= \text{True} \quad \text{Reach OVERRIDE state}
\end{align*}\]

Test case values:  
\[\begin{align*}
\text{Ignited} &= \text{False} \quad \text{P9 Triggering event} \\
\text{Ignited} &= \text{True} \quad \text{P1 Triggering event}
\end{align*}\]

Expected outputs: INACTIVE

5.3 Complete Sequence Level

At the complete sequence level, test engineers must use their experience and judgment to develop sequences of states that should be tested. To do this well requires experience with testing, experience with programming, and knowledge of the domain. These tests are omitted in this example.

5.4 Results

To evaluate this technique, a model of the cruise control problem was implemented in about 400 lines of C. The program accepts pairs of variable:values, where a value can be ‘t’, ‘f’, ‘T’, or ‘F’. Upper case inputs signify a triggering event. For convenience, the program was implemented so that the pre-state could be either set with a test case Prefix, or by an explicit input state value.

As a way to measure the quality of these tests, block and decision coverage was computed using the full predicate test cases. The coverage was measured using Atac [HI92]. The program, cruise, has five functions, 184 total blocks, and 174 decisions. The 54 test cases covered 163 of the blocks (89%) and 155 of the decisions (89%). Of the 20 uncovered decisions, five were infeasible, and eleven were related to input parameters that were not used during testing. That is, these eleven decisions were not related to the functional specifications given in Table 1. The remaining decisions were left uncovered because the variables Activate, Deactivate, and Resume are only used as triggering events, not condition variables. Although there have been very few published studies on the ability of specification-based tests to satisfy code-based coverage criteria, these results seem very promising.
6 INDUSTRIAL EXAMPLE TEST CASES

One of the goals of this project was to generate and provide examples of actual test cases that are derived from industrial software specifications. A specification was provided by Rockwell Collins, Inc. of a Flight Guidance System (FGS), written in CoRE [FBWK92]. No implementation has been provided, so the test case have not been applied.

The report supplied [MH97] describes the application CoRE method to the mode control logic of a Flight Guidance System. This example is considered to be a typical avionics problem, and complex enough to be realistic, but small enough to be used to evaluate software engineering techniques (such as the test data generation technique presented here).

Our test cases are derived primarily from the transition tables in Appendix A of the report. These tables are listed on page 11 of the report, under List of Tables. The tables are:

1. Overspeed Mode Transition Table
2. Autopilot Mode Transition Table
3. Autopilot ENGAGED Submode Transition Table
4. Autopilot DISENGAGED Submode Transition Table
5. Flight Director Mode Transition Table
6. Flight Director ON Transition Table
7. Active Lateral Mode Transition Table
8. Active Lateral ROLL Submode Transition Table
9. Active Lateral Navigation Submode Transition Table
10. Active Lateral Approach Submode Transition Table
11. Active Vertical Mode Transition Table
12. Flight Level Change Submode Transition Table
13. Altitude Select Mode Transition Table
14. Altitude Select ENABLED Submode Transition Table
15. Altitude Select ACTIVE Submode Transition Table
16. Vertical Approach Mode Transition Table
17. Vertical Approach ENABLED Submode Transition Table

The rest of this section provides each table and the associated tests. For many of the tables, variables were defined for several of the longer terms. This was done to save writing effort during construction of the test requirements. The definitions are given first, then the list of conditions. The conditions use the same syntax as in the cruise control example of Section 5; T or F represents a triggering event decision, @ represents a triggering event Boolean, t or f represents a WHEN condition, and a hyphen "-" represents a don’t care condition.

After the conditions, test requirements for full predicate coverage level test cases are given, then their associated test specifications. The transition pair coverage level test requirements are also
supplied. Complete sequence level tests are not included, as that requires domain knowledge that we do not have.

Four issues came up while these tests were being constructed. Two relate to handling of triggering events, another to test case prefixes, and a third that is specific to CoRE.

1. When these tests were started, the approach being used was to **not** modify trigger conditions during conjunctive level test generation. The experience of generating these tests convinced us that was wrong; and trigger conditions should be modified. This change was implemented in the cruise control case study of Section 5, and resulted in a higher level of branch coverage being achieved, but this change is not reflected in the following tests.

2. The triggering events were not adequately handled. Specifically, a triggering event includes an explicit value (the after-value), and an implicit before-value. To adequately test a triggering event, the before-values needed to be set. Again, this change was reflected in the cruise control case study, but not in the following tests.

3. During the generation of the FGS tests, the importance of prefixes to test cases were recognized. This was added to the technique, and the cruise control example, but is not reflected in the FGS tests.

4. CoRE allows nested submodes, which were not considered when this technique was being developed. The primary effect of this is on the transition-pair level tests. It is not clear whether it is reasonable to construct tests for transition-pairs when one mode is within a submode, and another is not. This is an issue that is left for future research, and as a result, the tests for transition tables 4, 9, 10, 15, 16, and 17 have no transition-pair requirements.

### 6.1 Overspeed Mode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SPEED_OK</td>
<td>@T(mon.Indicated_Airspeed &gt; (term.Vmo + 10 kts) AND NOT term.Above_Transition_Altitude)</td>
<td>TOO_FAST</td>
</tr>
<tr>
<td>2</td>
<td>SPEED_OK</td>
<td>@T(mon.Indicated_Mach_Number &gt; (term.Mmo + 0.03) AND term.Above_Transition_Altitude)</td>
<td>TOO_FAST</td>
</tr>
<tr>
<td>3</td>
<td>TOO_FAST</td>
<td>@T(mon.Indicated_Airspeed &lt;= term.Vmo AND NOT term.Above_Transition_Altitude)</td>
<td>SPEED_OK</td>
</tr>
<tr>
<td>4</td>
<td>TOO_FAST</td>
<td>@T(mon.Indicated_Mach_Number &lt;= term.Mmo AND term.Above_Transition_Altitude)</td>
<td>SPEED_OK</td>
</tr>
</tbody>
</table>

**Table 6.1: Overspeed Mode Transition Table**

(Table A.1, pg. 58 in the FGS report)

**Definitions:**

- X1 = (mon.Indicated_Airspeed > term.Vmo + 10kts)
- X2 = (mon.Indicated_Mach_Number > term.Mmo + 0.03)
- X3 = (mon.Indicated_Airspeed <= term.Vmo)
- X4 = (mon.Indicated_Mach_Number <= term.Mmo)
- X5 = term.Above_Transition_Altitude
• X1' — After-value of trigger event X1
• X2' — After-value of trigger event X2
• X3' — After-value of trigger event X3
• X4' — After-value of trigger event X4
• X5' — After-value of trigger event X5

6.1.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre X2 X3 X4 X5 X1' X2' X3' X4' X5' Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre State</td>
</tr>
<tr>
<td>P1 SPEED_OK  F   F   T   T   T   T   T   F   F</td>
</tr>
<tr>
<td>SPEED_OK    T   T   T   T   T   T   T   T   F</td>
</tr>
<tr>
<td>SPEED_OK    F   F   F   T   T   T   T   T   F</td>
</tr>
<tr>
<td>SPEED_OK    F   F   F   T   T   F   T   F   F</td>
</tr>
<tr>
<td>SPEED_OK    F   F   F   T   T   F   T   T   F</td>
</tr>
<tr>
<td>P2 SPEED_OK  -  F   T   T   T   T   F   F   F</td>
</tr>
<tr>
<td>SPEED_OK    -  T   T   T   T   T   F   F   F</td>
</tr>
<tr>
<td>SPEED_OK    -  F   T   T   T   F   F   F   F</td>
</tr>
<tr>
<td>P3 TOO_FAST  -  F   T   T   T   T   T   T   F</td>
</tr>
<tr>
<td>TOO_FAST    -  T   T   T   T   T   T   T   F</td>
</tr>
<tr>
<td>TOO_FAST    -  F   T   T   T   T   F   F   F</td>
</tr>
<tr>
<td>TOO_FAST    -  F   T   T   T   F   F   F   F</td>
</tr>
<tr>
<td>P4 TOO_FAST  -  -  -  -  -  -  -  -  -  T   T</td>
</tr>
<tr>
<td>TOO_FAST    -  -  -  -  -  -  -  -  -  T   T</td>
</tr>
<tr>
<td>TOO_FAST    -  -  -  -  -  -  -  -  -  T   T</td>
</tr>
<tr>
<td>TOO_FAST    -  -  -  -  -  -  -  -  -  T   T</td>
</tr>
</tbody>
</table>

Test specifications:

1) Test specification P1-1:

Prefix: X3 = True
X5 = False - Reach SPEED_OK
X4 = True
X5 = True - Reach SPEED_OK

Test case value: X1 = False - Trigger before-value
X5 = True - Trigger before-value
X1 = True - Trigger event
X5 = False - Trigger event

Expected Output: TOO_FAST

2) Test specification P1-2:

Prefix: X3 = True
X5 = False - Reach SPEED_OK
X4 = True  
X5 = True - Reach SPEED_OK
Test case value: X1 = True - Trigger before-value
X5 = True - Trigger before-value
X1 = True - Trigger event
X5 = False - Trigger event
Expected Output: SPEED_OK

3) Test specification P1-3:

Prefix: X3 = True
X5 = False - Reach SPEED_OK
X4 = True  
X5 = True - Reach SPEED_OK
Test case value: X1 = False - Trigger before-value
X5 = False - Trigger before-value
X1 = True - Trigger event
X5 = False - Trigger event
Expected Output: SPEED_OK

4) Test specification P1-4:

Prefix: X3 = True
X5 = False - Reach SPEED_OK
X4 = True  
X5 = True - Reach SPEED_OK
Test case value: X1 = False - Trigger before-value
X5 = False - Trigger before-value
X1 = True - Trigger event
X5 = False - Trigger event
Expected Output: SPEED_OK

5) Test specification P1-5:

Prefix: X3 = True
X5 = False - Reach SPEED_OK
X4 = True  
X5 = True - Reach SPEED_OK
Test case value: X1 = False - Trigger before-value
X5 = False - Trigger before-value
X1 = True - Trigger event
X5 = True - Trigger event
Expected Output: SPEED_OK

6) Test specification P2-1:

Prefix: X3 = True
X5 = False - Reach SPEED_OK
X4 = True  
X5 = True - Reach SPEED_OK
Test case value: \( X_2 = \text{False} \) - Trigger before-value
\( X_5 = \text{False} \) - Trigger before-value
\( X_2 = \text{True} \) - Trigger event
\( X_5 = \text{True} \) - Trigger event

Expected Output: TOO_FAST

7) Test specification P2-2:

Prefix:
\( X_3 = \text{True} \)
\( X_5 = \text{False} \) - Reach SPEED_OK
\( X_4 = \text{True} \)
\( X_5 = \text{True} \) - Reach SPEED_OK

Test case value:
\( X_2 = \text{True} \) - Trigger before-value
\( X_5 = \text{True} \) - Trigger before-value
\( X_2 = \text{True} \) - Trigger event
\( X_5 = \text{True} \) - Trigger event

Expected Output: SPEED_OK

8) Test specification P2-3:

Prefix:
\( X_3 = \text{True} \)
\( X_5 = \text{False} \) - Reach SPEED_OK
\( X_4 = \text{True} \)
\( X_5 = \text{True} \) - Reach SPEED_OK

Test case value:
\( X_2 = \text{False} \) - Trigger before-value
\( X_5 = \text{False} \) - Trigger before-value
\( X_2 = \text{False} \) - Trigger event
\( X_5 = \text{True} \) - Trigger event

Expected Output: SPEED_OK

9) Test specification P2-4:

Prefix:
\( X_3 = \text{True} \)
\( X_5 = \text{False} \) - Reach SPEED_OK
\( X_4 = \text{True} \)
\( X_5 = \text{True} \) - Reach SPEED_OK

Test case value:
\( X_2 = \text{False} \) - Trigger before-value
\( X_5 = \text{False} \) - Trigger before-value
\( X_2 = \text{False} \) - Trigger event
\( X_5 = \text{False} \) - Trigger event

Expected Output: SPEED_OK

10) Test specification P3-1:

Prefix:
\( X_1 = \text{True} \)
\( X_5 = \text{False} \) - Reach TOO_FAST
\( X_2 = \text{True} \)
\( X_5 = \text{True} \) - Reach TOO_FAST

Test case value:
\( X_3 = \text{False} \) - Trigger before-value
\( X_5 = \text{True} \) - Trigger before-value
\( X_3 = \text{True} \) - Trigger event
X5 = False - Trigger event
Expected Output: SPEED_OK

11) Test specification P3-2:

Prefix: X1 = True
        X5 = False - Reach TOO_FAST
        X2 = True
        X5 = True - Reach TOO_FAST
Test case value: X3 = True - Trigger before-value
                X5 = True - Trigger before-value
                X3 = True - Trigger event
                X5 = False - Trigger event
Expected Output: TOO_FAST

12) Test specification P3-3:

Prefix: X1 = True
        X5 = False - Reach TOO_FAST
        X2 = True
        X5 = True - Reach TOO_FAST
Test case value: X3 = False - Trigger before-value
                X5 = False - Trigger before-value
                X3 = True - Trigger event
                X5 = False - Trigger event
Expected Output: TOO_FAST

13) Test specification P3-4:

Prefix: X1 = True
        X5 = False - Reach TOO_FAST
        X2 = True
        X5 = True - Reach TOO_FAST
Test case value: X3 = False - Trigger before-value
                X5 = True - Trigger before-value
                X3 = False - Trigger event
                X5 = False - Trigger event
Expected Output: TOO_FAST

14) Test specification P3-5:

Prefix: X1 = True
        X5 = False - Reach TOO_FAST
        X2 = True
        X5 = True - Reach TOO_FAST
Test case value: X3 = False - Trigger before-value
                X5 = True - Trigger before-value
                X3 = True - Trigger event
                X5 = True - Trigger event
Expected Output: TOO_FAST
15) Test specification P4-1:

Prefix:  
X1 = True
X5 = False  - Reach TOO_FAST
X2 = True
X5 = True  - Reach TOO_FAST

Test case value:  
X4 = False - Trigger before-value
X5 = False - Trigger before-value
X4 = True - Trigger event
X5 = True - Trigger event

Expected Output: SPEED_OK

16) Test specification P4-2:

Prefix:  
X1 = True
X5 = False  - Reach TOO_FAST
X2 = True
X5 = True  - Reach TOO_FAST

Test case value:  
X4 = True - Trigger before-value
X5 = True - Trigger before-value
X4 = True - Trigger event
X5 = True - Trigger event

Expected Output: TOO_FAST

17) Test specification P4-3:

Prefix:  
X1 = True
X5 = False  - Reach TOO_FAST
X2 = True
X5 = True  - Reach TOO_FAST

Test case value:  
X4 = False - Trigger before-value
X5 = False - Trigger before-value
X4 = False - Trigger event
X5 = True - Trigger event

Expected Output: TOO_FAST

18) Test specification P4-4:

Prefix:  
X1 = True
X5 = False  - Reach TOO_FAST
X2 = True
X5 = True  - Reach TOO_FAST

Test case value:  
X4 = False - Trigger before-value
X5 = False - Trigger before-value
X4 = True - Trigger event
X5 = False - Trigger event

Expected Output: TOO_FAST

6.1.2 Transition Pair Coverage Level Requirements:

The pairs for the SPEED_OK Mode are:
The pairs for the TOO_FAST Mode are:

(P3 or P4) : (P1 or P2)

(P1 or P2) : (P3 or P4)

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>SPEED_OK:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>TOO_FAST</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>OR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>TOO_FAST</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>OR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>TOO_FAST</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>OR</td>
</tr>
</tbody>
</table>

Test specifications

1) Test specification SPEED_OK:
   Prefix:  X1 = True
   X5 = False - Reach TOO_FAST
   X2 = True
   X5 = True - Reach TOO_FAST
   Test case value:  X3 = True
   X5 = False - P3 trigger event
   X4 = True
   X5 = True - P4 trigger event
   X1 = True
   X5 = False - P1 trigger event
   X2 = True
   X5 = True - P2 trigger event
   X5 = True - P2 trigger event
   Expected Output: TOO_FAST

2) Test specification TOO_FAST:
   Prefix:  X3 = True
   X5 = False - Reach SPEED_OK
   X4 = True
   X5 = True - Reach SPEED_OK
   Test case value:  X1 = True
   X5 = False - P1 trigger event
   X2 = True
   X5 = True - P2 trigger event
   X3 = True
   X5 = False - P3 trigger event
X4 = True
X5 = True - P4 trigger event

Expected Output: SPEED_OK

### 6.2 Autopilot Mode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>DISENGAGED</td>
<td>@AP_Engage_Switch_Pressed WHEN mon_AP_Disconnect_Bar = UP</td>
<td>ENGAGED</td>
</tr>
<tr>
<td>6</td>
<td>ENGAGED</td>
<td>@AP_Disengage_Pressed</td>
<td>DISENGAGED</td>
</tr>
<tr>
<td>7</td>
<td>ENGAGED</td>
<td>@T(mon_AP_Disconnect_Bar = DOWN)</td>
<td>DISENGAGED</td>
</tr>
<tr>
<td>8</td>
<td>ENGAGED</td>
<td>@T(mon_AP_Disconnect_Bar = DOWN) OR mode_Active_Vertical = GA</td>
<td>DISENGAGED/Warning</td>
</tr>
</tbody>
</table>

Table 6.2: Autopilot Mode Transition Table
(Table A.2, pg. 63 in the FGS report)

Definitions:

- X1 = @(AP_Engage_Switch_Pressed)
- X2 = MON_AP_Disconnect_Bar = UP
- X3 = @(AP_Disengage_Pressed)
- X4 = (mon_AP_Disconnect_Bar = DOWN)
- X5 = (mode_Active_Lateral = GA)
- X6 = (mode_Active_Vertical = GA)
- X1' -- After-value of trigger event X1
- X3' -- After-value of trigger event X3
- X4' -- After-value of trigger event X4
- X5' -- After-value of trigger event X5
- X6' -- After-value of trigger event X6

### 6.2.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
<th>X1'</th>
<th>X3'</th>
<th>X4'</th>
<th>X5'</th>
<th>X6'</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P5 DISENGAGED NOT@ t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>ENGAGED</td>
</tr>
<tr>
<td>DISENGAGED @ t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>DISENGAGED</td>
</tr>
<tr>
<td>DISENGAGED NOT@ f</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>DISENGAGED</td>
</tr>
<tr>
<td>DISENGAGED NOT@ t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>NOT@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>DISENGAGED</td>
</tr>
<tr>
<td>P6 ENGAGED</td>
<td>-</td>
<td>-</td>
<td>NOT@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>DISENGAGED</td>
</tr>
<tr>
<td>ENGAGED</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>ENGAGED</td>
</tr>
<tr>
<td></td>
<td>ENGAGED</td>
<td>-</td>
<td>NOT</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>NOT</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>ENGAGED</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>---</td>
<td>-----</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>-----</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>P7</td>
<td>ENGAGED</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>DIENGAGED</td>
</tr>
<tr>
<td></td>
<td>ENGAGED</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>ENGAGED</td>
</tr>
<tr>
<td></td>
<td>ENGAGED</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>ENGAGED</td>
</tr>
<tr>
<td>P8</td>
<td>ENGAGED</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>T</td>
<td>DIENGAGED/Warning</td>
</tr>
<tr>
<td></td>
<td>ENGAGED</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>T</td>
<td>ENGAGED</td>
</tr>
<tr>
<td></td>
<td>ENGAGED</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>T</td>
<td>ENGAGED</td>
</tr>
<tr>
<td></td>
<td>ENGAGED</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>T</td>
<td>ENGAGED</td>
</tr>
<tr>
<td></td>
<td>ENGAGED</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>F</td>
<td>ENGAGED</td>
</tr>
</tbody>
</table>

Test specifications:

1) Test specification P5-1:

Prefix:  
X3 = True  - Reach DIENGAGED  
X4 = True  - Reach DIENGAGED  
X5 = True  - Reach DIENGAGED  
X6 = True  
Test case value:  
NOT X1  - Trigger before-value  
X2 = True  - Condition variable  
X1  - Trigger event  
Expected Output:  ENGAGED 

2) Test specification P5-2:

Prefix:  
X3  - Reach DIENGAGED  
X4 = True  - Reach DIENGAGED  
X5 = True  - Reach DIENGAGED  
X6 = True  
Test case value:  
X1  - Trigger before-value  
X2 = True  - Condition variable  
X1  - Trigger event  
Expected Output:  DIENGAGED 

3) Test specification P5-3:

Prefix:  
X3  - Reach DIENGAGED  
X4 = True  - Reach DIENGAGED  
X5 = True  - Reach DIENGAGED  
X6 = True  
Test case value:  
NOT X1  - Trigger before-value  
X2 = False  - Condition variable  
X1  - Trigger event  
Expected Output:  DIENGAGED 

4) Test specification P5-4:

Prefix:  
X3  - Reach DIENGAGED  
X4 = True  - Reach DIENGAGED  
X5 = True  - Reach DIENGAGED  
X6 = True  

35
Test case value: NOT X1 - Trigger before-value
X2 = True - Condition variable
NOT X1 - Trigger event
Expected Output: DISENGAGED

5) Test specification P6-1:
Prefix: X1 - Reach ENGAGED
X2 = True - Condition variable
Test case value: NOT X3 - Trigger before-value
X3 - Trigger event
Expected Output: DISENGAGED

6) Test specification P6-2:
Prefix: X1 - Reach ENGAGED
X2 = True - Condition variable
Test case value: X3 - Trigger before-value
X3 - Trigger event
Expected Output: ENGAGED

7) Test specification P6-3:
Prefix: X1 - Reach ENGAGED
X2 = True - Condition variable
Test case value: NOT X3 - Trigger before-value
NOT X3 - Trigger event
Expected Output: ENGAGED

8) Test specification P7-1:
Prefix: X1 - Reach ENGAGED
X2 = True - Condition variable
Test case value: X4 = False - Trigger before-value
X4 = True - Trigger event
Expected Output: DISENGAGED

9) Test specification P7-2:
Prefix: X1 - Reach ENGAGED
X2 = True - Condition variable
Test case value: X4 = True - Trigger before-value
X4 = True - Trigger event
Expected Output: ENGAGED

10) Test specification P7-3:
Prefix: X1 - Reach ENGAGED
X2 = True - Condition variable
Test case value: X4 = False - Trigger before-value
X4 = False - Trigger event
Expected Output: ENGAGED

11) Test specification P8-1:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>X1</th>
<th>- Reach ENGAGED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X2 = True</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>Test case value:</td>
<td>X5 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X6 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X5 = True</td>
<td>- Trigger event</td>
</tr>
<tr>
<td></td>
<td>X6 = True</td>
<td>- Trigger event</td>
</tr>
</tbody>
</table>

Expected Output: DISENGAGED/Warning

12) Test specification P8-2:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>X1</th>
<th>- Reach ENGAGED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X2 = True</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>Test case value:</td>
<td>X5 = True</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X6 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X5 = True</td>
<td>- Trigger event</td>
</tr>
<tr>
<td></td>
<td>X6 = True</td>
<td>- Trigger event</td>
</tr>
</tbody>
</table>

Expected Output: ENGAGED

13) Test specification P8-3:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>X1</th>
<th>- Reach ENGAGED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X2 = True</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>Test case value:</td>
<td>X5 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X6 = True</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X5 = True</td>
<td>- Trigger event</td>
</tr>
<tr>
<td></td>
<td>X6 = True</td>
<td>- Trigger event</td>
</tr>
</tbody>
</table>

Expected Output: ENGAGED

14) Test specification P8-4:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>X1</th>
<th>- Reach ENGAGED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X2 = True</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>Test case value:</td>
<td>X5 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X6 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X5 = False</td>
<td>- Trigger event</td>
</tr>
<tr>
<td></td>
<td>X6 = True</td>
<td>- Trigger event</td>
</tr>
</tbody>
</table>

Expected Output: ENGAGED

15) Test specification P8-5:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>X1</th>
<th>- Reach ENGAGED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X2 = True</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>Test case value:</td>
<td>X5 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X6 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X5 = True</td>
<td>- Trigger event</td>
</tr>
<tr>
<td></td>
<td>X6 = False</td>
<td>- Trigger event</td>
</tr>
</tbody>
</table>

Expected Output: ENGAGED
6.2.2 Transition Pair Coverage Level Requirements:

The pairs for the DISENGAGED Mode are:

\((P6 \text{ or } P7 \text{ or } P8) : P5\)

The pairs for the ENGAGED Mode are:

\(P5 : (P6 \text{ or } P7 \text{ or } P8)\)

<table>
<thead>
<tr>
<th></th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISENGAGED</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENGAGED</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENGAGED</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>ENGAGED</td>
<td>-</td>
<td>t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DIENGAGED</td>
<td>-</td>
<td>t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

\(\text{Test specifications}\)

1) Test specification DISENGAGED:
   Prefix: \(X_1\) - Reach ENGAGED
   Test case value: \(X_2 = \text{True}\) - Condition Variable
   \(X_3 = \text{True}\) - P6 trigger event
   \(X_4 = \text{True}\) - P7 trigger event
   \(X_5 = \text{True}\) - P8 trigger event
   \(X_6 = \text{True}\)
   \(X_1\) - P5 trigger event
   \(X_2 = \text{True}\) - Condition Variable
   Expected Output: ENGAGED

2) Test specification ENGAGED:
   Prefix: \(X_3\) - Reach DISENGAGED
   \(X_4 = \text{True}\) - Reach DISENGAGED
   \(X_5 = \text{True}\)
   \(X_6 = \text{True}\) - Reach DISENGAGED
   Test case value: \(X_1\) - P5 trigger event
   \(X_2 = \text{True}\) - Condition Variable
   \(\text{NOT} \ X_3\) - P6 trigger event
   \(X_4 = \text{True}\) - P7 trigger event
X5 = True - P8 trigger event
X6 = True
Expected Output: DIENGAGED

6.3 Aotopilot ENGAGED Submode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>Normal</td>
<td>@T(term_SYNC)</td>
<td>Sync</td>
</tr>
<tr>
<td>10</td>
<td>Sync</td>
<td>@F(term_SYNC)</td>
<td>Normal</td>
</tr>
</tbody>
</table>

Table 6.3: Aotopilot ENGAGED Submode Transition Table
(Table A.3, pg. 63 in the FGS report)

6.3.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>term_SYNC</th>
<th>term_SYNC'</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P9</td>
<td>Normal</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>Normal</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>Normal</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>P10</td>
<td>Sync</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>Sync</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>Sync</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>

Test specifications:

1) Test specification P9-1:

Prefix:  term_SYNC = False - Reach Normal
Test case value: term_SYNC = False - Trigger before-value
                term_SYNC = True  - Trigger event
Expected Output: Sync

2) Test specification P9-2:

Prefix:  term_SYNC = False - Reach Normal
Test case value: term_SYNC = True  - Trigger before-value
                term_SYNC = True  - Trigger event
Expected Output: Normal

3) Test specification P9-3:

Prefix:  term_SYNC = False - Reach Normal
Test case value: term_SYNC = False - Trigger before-value
                term_SYNC = False - Trigger event
Expected Output: Normal

4) Test specification P10-1:

Prefix:  term_SYNC = True - Reach Sync
Test case value: term_SYNC = True  - Trigger before-value
term_SYNC = False  - Trigger event

Expected Output: Normal

5) Test specification P10-2:

Prefix:  term_SYNC = True  - Reach Sync
Test case value: term_SYNC = False  - Trigger before-value
     term_SYNC = False  - Trigger event

Expected Output: Sync

6) Test specification P10-3:

Prefix:  term_SYNC = True  - Reach Sync
Test case value: term_SYNC = True  - Trigger before-value
     term_SYNC = True  - Trigger event

Expected Output: Sync

6.3.2 Transition Pair Coverage Level Requirements:

The pairs for the Normal Mode are:

P10 : P9

The pairs for the Sync Mode are:

P9 : P10

<table>
<thead>
<tr>
<th>term_SYNC'</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
</tr>
<tr>
<td>Normal</td>
</tr>
<tr>
<td>Sync</td>
</tr>
<tr>
<td>Sync</td>
</tr>
</tbody>
</table>

Test specifications

1) Test specification Normal:

Prefix:  term_Sync = True  - Reach Sync
Test case value: term_Sync = False  - P10 trigger event
     term_Sync = True  - P9 trigger event

Expected Output: Sync

2) Test specification Sync:

Prefix:  term_Sync = False  - Reach Normal
Test case value: term_Sync = True  - P9 trigger event
     term_Sync = False  - P10 trigger event

Expected Output: Normal
6.4 Autopilot DISENGAGED Submode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>Warning</td>
<td>@T(Duration(INMODE(WARNING)) &gt; 10 sec)</td>
<td>Normal</td>
</tr>
</tbody>
</table>

Table 6.4: Autopilot DISENGAGED Submode Transition Table
(Table A.4, pg. 63 in the FGS report)

Definitions:

- $X_1 = \text{Duration(INMODE(WARNING))} > 10 \text{ sec}$
- $X' = \text{After-value of trigger event } X_1$
- $\text{Pre-P10} = (\text{mode_Active_Lateral} = \text{GA or mode_Active_Vertical} = \text{GA})$

6.4.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>X1</th>
<th>X1’</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P10</td>
<td>Warning</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>Warning</td>
<td>T</td>
<td>T</td>
<td>Warning</td>
</tr>
<tr>
<td>Warning</td>
<td>F</td>
<td>F</td>
<td>Warning</td>
</tr>
</tbody>
</table>

Test specifications:

1) Test specification P11-1:

Prefix: Pre-P10 = True - Reach Warning
Test case value: X1 = False - Trigger before-value
X1 = True - Trigger event
Expected Output: Normal

2) Test specification P11-2:

Prefix: Pre-P10 = True - Reach Warning
Test case value: X1 = True - Trigger before-value
X1 = True - Trigger event
Expected Output: Warning

3) Test specification P11-3:

Prefix: Pre-P10 = True - Reach Warning
Test case value: X1 = True - Trigger before-value
X1 = True - Trigger event
Expected Output: Warning

6.4.2 Transition Pair Coverage Level Requirements:

NONE.
6.5 Flight Director Mode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>OFF</td>
<td>@Flight_Mode_Requested</td>
<td>ON</td>
</tr>
<tr>
<td>13</td>
<td>OFF</td>
<td>@T(term_Overspeed)</td>
<td>OFF</td>
</tr>
<tr>
<td>14</td>
<td>OFF</td>
<td>@T(term_AP_Engaged)</td>
<td>OFF</td>
</tr>
<tr>
<td>15</td>
<td>OFF</td>
<td>@FD_Pressed</td>
<td>OFF</td>
</tr>
<tr>
<td>16</td>
<td>ON</td>
<td>@FD_Pressed WHEN (NOT term_Overspeed AND NOT term_AP_Engaged)</td>
<td>OFF</td>
</tr>
</tbody>
</table>

Table 6.5: Flight Director Mode Transition Table
(Table A.5, pg. 69 in the FGS report)

Definitions:
- X1 = @Flight_Mode_Requested
- X2 = term_Overspeed
- X3 = term_AP_Engaged
- X4 = @FD_Pressed
- X1' -- After-value of trigger event X1
- X2' -- After-value of trigger event X2
- X3' -- After-value of trigger event X3
- X4' -- After-value of trigger event X4

6.5.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>Pre X1</th>
<th>Pre X2</th>
<th>Pre X3</th>
<th>Pre X4</th>
<th>Pre X1'</th>
<th>Pre X2'</th>
<th>Pre X3'</th>
<th>Pre X4'</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P12</td>
<td>OFF</td>
<td>NOT@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>NOT@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>NOT@</td>
<td>-</td>
<td>-</td>
<td>OFF</td>
</tr>
<tr>
<td>P13</td>
<td>OFF</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>OFF</td>
</tr>
<tr>
<td>P14</td>
<td>OFF</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>OFF</td>
</tr>
<tr>
<td>P15</td>
<td>OFF</td>
<td>-</td>
<td>-</td>
<td>NOT@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>-</td>
<td>-</td>
<td>NOT@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>NOT@</td>
<td>OFF</td>
</tr>
<tr>
<td>P16</td>
<td>ON</td>
<td>-</td>
<td>-</td>
<td>NOT@</td>
<td>-</td>
<td>f</td>
<td>f</td>
<td>@</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>ON</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>f</td>
<td>f</td>
<td>@</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>ON</td>
<td>-</td>
<td>-</td>
<td>NOT@</td>
<td>t</td>
<td>f</td>
<td>@</td>
<td>ON</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ON</td>
<td>-</td>
<td>-</td>
<td>NOT@</td>
<td>-</td>
<td>f</td>
<td>t</td>
<td>@</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>ON</td>
<td>-</td>
<td>-</td>
<td>NOT@</td>
<td>-</td>
<td>t</td>
<td>f</td>
<td>NOT@</td>
<td>ON</td>
</tr>
</tbody>
</table>

42
Test specifications:

1) Test specification P12-1:

Prefix: X4 - Reach OFF
X2 = False - Condition variable
X3 = False - Condition variable
Test case value: NOT X1 - Trigger before-value
X1 - Trigger event
 Expected Output: ON

2) Test specification P12-2:

Prefix: X4 - Reach OFF
X2 = False - Condition variable
X3 = False - Condition variable
Test case value: X1 - Trigger before-value
X1 - Trigger event
 Expected Output: OFF

3) Test specification P12-3:

Prefix: X4 - Reach OFF
X2 = False - Condition variable
X3 = False - Condition variable
Test case value: NOT X1 - Trigger before-value
NOT X1 - Trigger event
 Expected Output: OFF

4) Test specification P13-1:

Prefix: X4 - Reach OFF
X2 = False - Condition variable
X3 = False - Condition variable
Test case value: X2 = False - Trigger before-value
X2 = True - Trigger event
 Expected Output: ON

5) Test specification P13-2:

Prefix: X4 - Reach OFF
X2 = False - Condition variable
X3 = False - Condition variable
Test case value: X2 = True - Trigger before-value
X2 = True - Trigger event
 Expected Output: OFF

6) Test specification P13-3:

Prefix: X4 - Reach OFF
X2 = False - Condition variable
X3 = False - Condition variable
Test case value: \( X_2 = \text{False} \) - Trigger before-value
\( X_2 = \text{False} \) - Trigger event
Expected Output: OFF

7) Test specification P14-1:

Prefix: \( X_4 \) - Reach OFF
\( X_2 = \text{False} \) - Condition variable
\( X_3 = \text{False} \) - Condition variable
Test case value: \( X_3 = \text{False} \) - Trigger before-value
\( X_3 = \text{True} \) - Trigger event
Expected Output: ON

8) Test specification P14-2:

Prefix: \( X_4 \) - Reach OFF
\( X_2 = \text{False} \) - Condition variable
\( X_3 = \text{False} \) - Condition variable
Test case value: \( X_3 = \text{True} \) - Trigger before-value
\( X_3 = \text{True} \) - Trigger event
Expected Output: OFF

9) Test specification P14-3:

Prefix: \( X_4 \) - Reach OFF
\( X_2 = \text{False} \) - Condition variable
\( X_3 = \text{False} \) - Condition variable
Test case value: \( X_3 = \text{False} \) - Trigger before-value
\( X_3 = \text{False} \) - Trigger event
Expected Output: OFF

10) Test specification P15-1:

Prefix: \( X_4 \) - Reach OFF
\( X_2 = \text{False} \) - Condition variable
\( X_3 = \text{False} \) - Condition variable
Test case value: \( \text{NOT} X_4 \) - Trigger before-value
\( X_4 \) - Trigger event
Expected Output: ON

11) Test specification P15-2:

Prefix: \( X_4 \) - Reach OFF
\( X_2 = \text{False} \) - Condition variable
\( X_3 = \text{False} \) - Condition variable
Test case value: \( X_4 \) - Trigger before-value
\( X_4 \) - Trigger event
Expected Output: OFF

12) Test specification P15-3:
13) Test specification P16-1:

Prefix: X4 - Reach OFF
X2 = False - Condition variable
X3 = False - Condition variable
Test case value: NOT X4 - Trigger before-value
NOT X4 - Trigger event
Expected Output: OFF

14) Test specification P16-2:

Prefix: X1 - Reach ON
X2 = True - Reach ON
X3 = True - Reach ON
Test case value: NOT X4 - Trigger before-value
X2 = False - Condition variable
X3 = False - Condition variable
X4 - Trigger event
Expected Output: OFF

15) Test specification P16-3:

Prefix: X1 - Reach ON
X2 = True - Reach ON
X3 = True - Reach ON
Test case value: NOT X4 - Trigger before-value
X2 = True - Condition variable
X3 = False - Condition variable
X4 - Trigger event
Expected Output: ON

16) Test specification P16-4:

Prefix: X1 - Reach ON
X2 = True - Reach ON
X3 = True - Reach ON
Test case value: NOT X4 - Trigger before-value
X2 = False - Condition variable
X3 = True - Condition variable
X4 - Trigger event
Expected Output: ON
17) Test specification P16-5:

Prefix: 
- Reach ON
X2 = True - Reach ON
X3 = True - Reach ON

Test case value: 
- Trigger before-value
X2 = False - Condition variable
X3 = False - Condition variable
NOT X4 - Trigger event

Expected Output: ON

6.5.2 Transition Pair Coverage Level Requirements:

The pairs for the OFF Mode are:

P16 : (P12 or P13 or P14 or P15)

The pairs for the ON Mode are:

(P12 or P13 or P14 or P15) : P16

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>OFF</th>
<th>ON</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>@</td>
<td></td>
<td></td>
<td></td>
<td>ON</td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td>T</td>
<td></td>
<td></td>
<td>ON</td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td></td>
<td>T</td>
<td>@</td>
<td>ON</td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td></td>
<td>@</td>
<td></td>
<td>ON</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
<td>@</td>
<td></td>
<td></td>
<td>ON</td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td>T</td>
<td></td>
<td></td>
<td>ON</td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td></td>
<td>T</td>
<td>@</td>
<td>ON</td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td></td>
<td>@</td>
<td></td>
<td>ON</td>
</tr>
</tbody>
</table>

Test specifications

1) Test specification OFF:

Prefix: 
- Reach ON
X2 = True - Reach ON
X3 = True - Reach ON
X4 - Reach ON

Test case value: X4 - P16 trigger event
\[ X_2 = \text{False} \quad \text{- Condition variable} \]
\[ X_3 = \text{False} \quad \text{- Condition variable} \]
\[ X_1 \quad \text{- P12 trigger event} \]
\[ X_2 = \text{True} \quad \text{- P13 trigger event} \]
\[ X_3 = \text{True} \quad \text{- P14 trigger event} \]
\[ X_4 \quad \text{- P15 trigger event} \]

Expected Output: ON

2) Test specification ON:
Prefix: 
\[ X_4 \quad \text{- Reach OFF} \]
\[ X_2 = \text{False} \quad \text{- Condition variable} \]
\[ X_3 = \text{False} \quad \text{- Condition variable} \]

Test case value: 
\[ X_1 \quad \text{- P12 trigger event} \]
\[ X_2 = \text{True} \quad \text{- P13 trigger event} \]
\[ X_3 = \text{True} \quad \text{- P14 trigger event} \]
\[ X_4 \quad \text{- P15 trigger event} \]
\[ X_4 \quad \text{- P16 trigger event} \]
\[ X_2 = \text{False} \quad \text{- Condition variable} \]
\[ X_3 = \text{False} \quad \text{- Condition variable} \]

Expected Output: OFF

6.6 Flight Director ON Submode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>No_Cues</td>
<td>@FD_Pressed WHEN (term_AP_Engaged OR term_Overspeed)</td>
<td>Cues</td>
</tr>
<tr>
<td>18</td>
<td>No_Cues</td>
<td>@T(term_Overspeed)</td>
<td>Cues</td>
</tr>
<tr>
<td>19</td>
<td>Cues</td>
<td>@FD_Pressed WHEN (term_AP_Engaged OR term_Overspeed)</td>
<td>No_Cues</td>
</tr>
</tbody>
</table>

Table 6.6: Flight Director ON Submode Transition Table

(Table A.6, pg. 69 in the FGS report)

Definitions:
- \( X_1 = \text{@FD_Pressed} \)
- \( X_2 = \text{term_AP_Engaged} \)
- \( X_3 = \text{term_Overspeed} \)
- \( X_1' = \text{After-value of trigger event } X_1 \)
- \( X_3' = \text{After-value of trigger event } X_3 \)

6.6.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X1'</th>
<th>X3'</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P17</td>
<td>No_Cues</td>
<td>NOT@</td>
<td>t</td>
<td>-</td>
<td>@</td>
<td>Cues</td>
</tr>
<tr>
<td></td>
<td>No_Cues</td>
<td>@</td>
<td>t</td>
<td>-</td>
<td>@</td>
<td>No_Cues</td>
</tr>
<tr>
<td></td>
<td>No_Cues</td>
<td>NOT@</td>
<td>f</td>
<td>-</td>
<td>@</td>
<td>No_Cues</td>
</tr>
</tbody>
</table>
Test specifications:

1) Test specification P17-1:

Prefix: X1 - Reach No_Cues
X2 = True - Condition variable
X3 = True - Condition variable
Test case value: NOT X1 - Trigger before-value
X2 = True - Condition variable
X1 - Trigger event
Expected Output: Cues

2) Test specification P17-2:

Prefix: X1 - Reach No_Cues
X2 = True - Condition variable
X3 = True - Condition variable
Test case value: X1 - Trigger before-value
X2 = True - Condition variable
X1 - Trigger event
Expected Output: No_Cues

3) Test specification P17-3:

Prefix: X1 - Reach No_Cues
X2 = True - Condition variable
X3 = True - Condition variable
Test case value: NOT X1 - Trigger before-value
X2 = False - Condition variable
X1 - Trigger event
Expected Output: No_Cues

4) Test specification P17-4:

Prefix: X1 - Reach No_Cues
X2 = True - Condition variable
X3 = True - Condition variable
Test case value: NOT X1 - Trigger before-value
X2 = True - Condition variable
NOT X1 - Trigger event
Expected Output: No_Cues

5) Test specification P17-5:
Prefix: X1 - Reach No_Cues
X2 = True - Condition variable
X3 = True - Condition variable
Test case value: NOT X1 - Trigger before-value
X3 = True - Condition variable
X1 - Trigger event
Expected Output: Cues

6) Test specification P17-6:
Prefix: X1 - Reach No_Cues
X2 = True - Condition variable
X3 = True - Condition variable
Test case value: X1 - Trigger before-value
X3 = True - Condition variable
X1 - Trigger event
Expected Output: No_Cues

7) Test specification P17-7:
Prefix: X1 - Reach No_Cues
X2 = True - Condition variable
X3 = True - Condition variable
Test case value: NOT X1 - Trigger before-value
X3 = False - Condition variable
X1 - Trigger event
Expected Output: No_Cues

8) Test specification P17-8:
Prefix: X1 - Reach No_Cues
X2 = True - Condition variable
X3 = True - Condition variable
Test case value: NOT X1 - Trigger before-value
X3 = True - Condition variable
NOT X1 - Trigger event
Expected Output: No_Cues

9) Test specification P18-1:
Prefix: X1 - Reach No_Cues
X2 = True - Condition variable
Test case value: X3 = False - Trigger before-value
X3 = True - Trigger event
Expected Output: Cues

10) Test specification P18-2:

Prefix: X1 - Reach No_Cues
X2 = True - Condition variable
Test case value: X3 = True - Trigger before-value
X3 = False - Trigger event
Expected Output: No_Cues

11) Test specification P18-3:

Prefix: X1 - Reach No_Cues
X2 = True - Condition variable
Test case value: X3 = True - Trigger before-value
X3 = False - Trigger event
Expected Output: No_Cues

12) Test specification P19-1:

Prefix: X2 = True - Condition variable
X3 = True - Condition variable
X1 - Reach Cues
X3 = True - Reach Cues
Test case value: NOT X1 - Trigger before-value
X2 = True - Condition variable
X1 - Trigger event
Expected Output: No_Cues

13) Test specification P19-2:

Prefix: X2 = True - Condition variable
X3 = True - Condition variable
X1 - Reach Cues
X3 = True - Reach Cues
Test case value: X1 - Trigger before-value
X2 = True - Condition variable
X1 - Trigger event
Expected Output: Cues

14) Test specification P19-3:

Prefix: X2 = True - Condition variable
X3 = True - Condition variable
X1 - Reach Cues
X3 = True - Reach Cues
Test case value: NOT X1 - Trigger before-value
X2 = False - Condition variable
X1 - Trigger event
Expected Output: Cues

15) Test specification P19-4:

Prefix: X2 = True - Condition variable
X3 = True - Condition variable
X1 - Reach Cues
X3 = True - Reach Cues
Test case value: NOT X1 - Trigger before-value
X2 = True - Condition variable
NOT X1 - Trigger event
Expected Output: Cues

16) Test specification P19-5:

Prefix: X2 = True - Condition variable
X3 = True - Condition variable
X1 - Reach Cues
X3 = True - Reach Cues
Test case value: NOT X1 - Trigger before-value
X3 = True - Condition variable
X1 - Trigger event
Expected Output: No_Cues

17) Test specification P19-6:

Prefix: X2 = True - Condition variable
X3 = True - Condition variable
X1 - Reach Cues
X3 = True - Reach Cues
Test case value: X1 - Trigger before-value
X3 = True - Condition variable
X1 - Trigger event
Expected Output: Cues

18) Test specification P19-7:

Prefix: X2 = True - Condition variable
X3 = True - Condition variable
X1 - Reach Cues
X3 = True - Reach Cues
Test case value: NOT X1 - Trigger before-value
X3 = False - Condition variable
X1 - Trigger event
Expected Output: Cues

19) Test specification P19-8:

Prefix: X2 = True - Condition variable
X3 = True - Condition variable
X1 - Reach Cues
X3 = True - Reach Cues
Test case value: NOT X1 - Trigger before-value
X3 = True - Condition variable
NOT X1 - Trigger event
Expected Output: Cues

6.6.2 Transition Pair Coverage Level Requirements:

The pairs for the No_Cues Mode are:

P19 : (P17 or P18)

The pairs for the Cues Mode are:

(P17 or P18) : P19

<table>
<thead>
<tr>
<th></th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
</tr>
</thead>
<tbody>
<tr>
<td>No_Cues</td>
<td>Cues</td>
<td>@</td>
<td>t</td>
</tr>
<tr>
<td>Cues</td>
<td>Cues</td>
<td>@</td>
<td>-</td>
</tr>
<tr>
<td>No_Cues</td>
<td>@</td>
<td>t</td>
<td>-</td>
</tr>
<tr>
<td>No_Cues</td>
<td>@</td>
<td>-</td>
<td>t</td>
</tr>
<tr>
<td>OR</td>
<td>No_Cues</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Cues</td>
<td>No_Cues</td>
<td>@</td>
<td>t</td>
</tr>
<tr>
<td>No_Cues</td>
<td>@</td>
<td>-</td>
<td>t</td>
</tr>
<tr>
<td>OR</td>
<td>No_Cues</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Cues</td>
<td>@</td>
<td>t</td>
<td>-</td>
</tr>
<tr>
<td>Cues</td>
<td>@</td>
<td>-</td>
<td>t</td>
</tr>
</tbody>
</table>

Test specifications

1) Test specification No_Cues:

Prefix:
X2 = True - Condition Variable
X3 = True - Condition Variable
X1 - Reach Cues
X3 - Reach Cues

Test case value:
X1 - P19 trigger event
X2 = True - Condition variable
X3 = True - Condition variable
X1 - P17 trigger event
X2 = True - Condition variable
X3 = True - Condition variable
X3 = True - P18 trigger event
Expected Output: Cues

2) Test specification Cues:

Prefix:  
X2 = True - Condition Variable  
X3 = True - Condition Variable  
X1 - Reach No_Cues

Test case value:  
X1 - P17 trigger event  
X2 = True - Condition variable  
X3 = True - Condition variable  
X3 = True - P18 trigger event  
X1 - P19 trigger event  
X2 = True - Condition variable  
X3 = True - Condition variable

Expected Output: Cues

6.7 Active Lateral Mode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>HDG</td>
<td>@HDG_Switch_Pressed</td>
<td>ROLL</td>
</tr>
<tr>
<td>21</td>
<td>NAV</td>
<td>@NAV_Switch_Pressed</td>
<td>ROLL</td>
</tr>
<tr>
<td>22</td>
<td>NAV</td>
<td>@Nav_Source_Change</td>
<td>ROLL</td>
</tr>
<tr>
<td>23</td>
<td>APPR</td>
<td>@APPR_Switch_Pressed</td>
<td>ROLL</td>
</tr>
<tr>
<td>24</td>
<td>APPR</td>
<td>@Nav_Source_Change</td>
<td>ROLL</td>
</tr>
<tr>
<td>25</td>
<td>GA</td>
<td>@T(term_AP_Engaged)</td>
<td>ROLL</td>
</tr>
<tr>
<td>26</td>
<td>GA</td>
<td>@T(term_SYNC)</td>
<td>ROLL</td>
</tr>
<tr>
<td>27</td>
<td>GA</td>
<td>@F(mode_Active_Vertical = GA)</td>
<td>ROLL</td>
</tr>
<tr>
<td>28</td>
<td>HDG</td>
<td>@HDG_Switch_Pressed</td>
<td>HDG</td>
</tr>
<tr>
<td>29</td>
<td>NAV</td>
<td>@NAV_Switch_Pressed</td>
<td>NAV</td>
</tr>
<tr>
<td>30</td>
<td>APPR</td>
<td>@APPR_Switch_Pressed</td>
<td>APPR</td>
</tr>
<tr>
<td>31</td>
<td>GA</td>
<td>@GA_Switch_Pressed</td>
<td>GA</td>
</tr>
</tbody>
</table>

Table 6.7: Active Lateral Mode Transition Table  
(Table A.7, pg. 75 in the FGS report)

Definitions:

- X1 = @HDG_Switch_Pressed
- X2 = @NAV_Switch_Pressed
- X3 = @Nav_Source_Change
- X4 = @APPR_Switch_Pressed
- X5 = term_AP_Engaged
- X6 = term_SYNC
- X7 = (mode_Active_Vertical = GA)
- X8 = GA_Switch_Pressed
- X1' = After-value of trigger event X1
- X2' --- After-value of trigger event X2
- X3' --- After-value of trigger event X3
- X4' --- After-value of trigger event X4
- X5' --- After-value of trigger event X5
- X6' --- After-value of trigger event X6
- X7' --- After-value of trigger event X7
- X8' --- After-value of trigger event X8

6.7.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
<th>X7</th>
<th>X8</th>
<th>X1' X2' X3' X4' X5' X6' X7' X8'</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P20</td>
<td>HDG NOT@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>HDG @</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>HDG NOT@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>P21</td>
<td>NAV - NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>NAV - @</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>NAV - NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>P22</td>
<td>NAV - NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>NAV - @</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>NAV - NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>P23</td>
<td>APPR - NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>APPR - @</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>APPR - NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>P24</td>
<td>APPR - NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>APPR - @</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>APPR - NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>P25</td>
<td>GA - F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>GA T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>GA F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>P26</td>
<td>GA T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>GA F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>GA T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>P27</td>
<td>GA T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>GA F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>GA T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>P28</td>
<td>NAV NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>NAV @</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>NAV NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>APPR NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>APPR @</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>APPR NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>GA NOT@-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>GA @</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>GA NOT-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Test specifications:

1) Test specification P20-1:

Prefix: X1
Test case value: NOT X1, X1
Expected Output: ROLL

2) Test specification P20-2:

Prefix: X1
Test case value: X1
Expected Output: HDG

3) Test specification P20-3:

Prefix: X1
Test case value: NOT X1, X1
Expected Output: HDG

4) Test specification P21-1:
Prefix: X2
Test case value: NOT X2
X2
Expected Output: ROLL

5) Test specification P21-2:
Prefix: X2
Test case value: X2
X2
Expected Output: NAV

6) Test specification P21-3:
Prefix: X2
Test case value: NOT X2
NOT X2
Expected Output: NAV

7) Test specification P22-1:
Prefix: X2
Test case value: NOT X3
X3
Expected Output: ROLL

8) Test specification P22-2:
Prefix: X2
Test case value: X3
X3
Expected Output: NAV

9) Test specification P22-3:
Prefix: X2
Test case value: NOT X3
NOT X3
Expected Output: NAV

10) Test specification P23-1:
Prefix: X4
Test case value: NOT X4
X4
Expected Output: ROLL

11) Test specification P23-2:
Prefix: X4
Test case value: X4
- Reach APPR
- Trigger event before value
12) Test specification P23-3:

Prefix: X4 - Reach APPR
Test case value: NOT X4 - Trigger event before value
NOT X4 - Trigger event
Expected Output: APPR

13) Test specification P24-1:

Prefix: X4 - Reach APPR
Test case value: NOT X3 - Trigger event before value
X3 - Trigger event
Expected Output: ROLL

14) Test specification P24-2:

Prefix: X4 - Reach APPR
Test case value: X3 - Trigger event before value
X3 - Trigger event
Expected Output: APPR

15) Test specification P24-3:

Prefix: X4 - Reach APPR
Test case value: NOT X3 - Trigger event before value
NOT X3 - Trigger event
Expected Output: APPR

16) Test specification P25-1:

Prefix: X8 - Reach GA
Test case value: X5 = False - Trigger event before-value
X5 = True - Trigger event
Expected Output: ROLL

17) Test specification P25-2:

Prefix: X8 - Reach GA
Test case value: X5 = True - Trigger event before-value
X5 = True - Trigger event
Expected Output: GA

18) Test specification P25-3:

Prefix: X8 - Reach GA
Test case value: X5 = False - Trigger event before-value
X5 = False - Trigger event
Expected Output: GA
19) Test specification P26-1:

Prefix: $X_8$ - Reach GA
Test case value: $X_6 = \text{False}$ - Trigger event before-value
$X_6 = \text{True}$ - Trigger event
Expected Output: ROLL

20) Test specification P26-2:

Prefix: $X_8$ - Reach GA
Test case value: $X_6 = \text{True}$ - Trigger event before-value
$X_6 = \text{True}$ - Trigger event
Expected Output: GA

21) Test specification P26-3:

Prefix: $X_8$ - Reach GA
Test case value: $X_6 = \text{False}$ - Trigger event before-value
$X_6 = \text{False}$ - Trigger event
Expected Output: GA

22) Test specification P27-1:

Prefix: $X_8$ - Reach GA
Test case value: $X_7 = \text{True}$ - Trigger event before-value
$X_7 = \text{False}$ - Trigger event
Expected Output: ROLL

23) Test specification P27-2:

Prefix: $X_8$ - Reach GA
Test case value: $X_7 = \text{False}$ - Trigger event before-value
$X_7 = \text{False}$ - Trigger event
Expected Output: GA

24) Test specification P27-3:

Prefix: $X_8$ - Reach GA
Test case value: $X_7 = \text{True}$ - Trigger event before-value
$X_7 = \text{True}$ - Trigger event
Expected Output: GA

25) Test specification P28-1:

Prefix: $X_2$ - Reach NAV
Test case value: $\text{NOT } X_1$ - Trigger event before-value
$X_1$ - Trigger event
Expected Output: HDG

26) Test specification P28-2:
Prefix: X2
Test case value: X1
- Reach NAV
- Trigger event before-value
- Trigger event

Expected Output: NAV

27) Test specification P28-3:

Prefix: X2
Test case value: NOT X1
- Reach NAV
- Trigger event before-value
- Trigger event

Expected Output: NAV

28) Test specification P28-4:

Prefix: X4
Test case value: NOT X1
- Reach APPR
- Trigger event before-value
- Trigger event

Expected Output: HDG

29) Test specification P28-5:

Prefix: X4
Test case value: X1
- Reach APPR
- Trigger event before-value
- Trigger event

Expected Output: APPR

30) Test specification P28-6:

Prefix: X4
Test case value: NOT X1
- Reach APPR
- Trigger event before-value
- Trigger event

Expected Output: APPR

31) Test specification P28-7:

Prefix: X8
Test case value: NOT X1
- Reach GA
- Trigger event before-value
- Trigger event

Expected Output: HDG

32) Test specification P28-8:

Prefix: X8
Test case value: X1
- Reach GA
- Trigger event before-value
- Trigger event

Expected Output: GA

33) Test specification P28-9:

Prefix: X8
Test case value: NOT X1
- Reach GA
- Trigger event before-value
NOT X1 - Trigger event
Expected Output: GA

34) Test specification P29-1:
Prefix: X1 - Reach HDG
Test case value: NOT X2 - Trigger event before-value
X2 - Trigger event
Expected Output: NAV

35) Test specification P29-2:
Prefix: X1 - Reach HDG
Test case value: X2 - Trigger event before-value
X2 - Trigger event
Expected Output: HDG

36) Test specification P29-3:
Prefix: X1 - Reach HDG
Test case value: NOT X2 - Trigger event before-value
NOT X2 - Trigger event
Expected Output: HDG

37) Test specification P29-4:
Prefix: X4 - Reach APPR
Test case value: NOT X2 - Trigger event before-value
X2 - Trigger event
Expected Output: NAV

38) Test specification P29-5:
Prefix: X4 - Reach APPR
Test case value: X2 - Trigger event before-value
X2 - Trigger event
Expected Output: APPR

39) Test specification P29-6:
Prefix: X4 - Reach APPR
Test case value: NOT X2 - Trigger event before-value
NOT X2 - Trigger event
Expected Output: APPR

40) Test specification P29-7:
Prefix: X8 - Reach GA
Test case value: NOT X2 - Trigger event before-value
X2 - Trigger event
Expected Output: NAV
41) Test specification P29-8:

Prefix: X8  - Reach GA
Test case value: X2  - Trigger event before-value
                X2  - Trigger event
Expected Output: GA

42) Test specification P29-9:

Prefix: X8  - Reach GA
Test case value: NOT X2  - Trigger event before-value
                NOT X2  - Trigger event
Expected Output: GA

43) Test specification P30-1:

Prefix: X1  - Reach HDG
Test case value: NOT X4  - Trigger event before-value
                X4  - Trigger event
Expected Output: APPR

44) Test specification P30-2:

Prefix: X1  - Reach HDG
Test case value: X4  - Trigger event before-value
                X4  - Trigger event
Expected Output: HDG

45) Test specification P30-3:

Prefix: X1  - Reach HDG
Test case value: NOT X4  - Trigger event before-value
                NOT X4  - Trigger event
Expected Output: HDG

46) Test specification P30-4:

Prefix: X2  - Reach NAV
Test case value: NOT X4  - Trigger event before-value
                X4  - Trigger event
Expected Output: APPR

47) Test specification P30-5:

Prefix: X2  - Reach NAV
Test case value: X4  - Trigger event before-value
                X4  - Trigger event
Expected Output: NAV

48) Test specification P30-6:
49) Test specification P30-7:

Prefix: X2 - Reach NAV
Test case value: NOT X4 - Trigger event before-value
NOT X4 - Trigger event
Expected Output: NAV

50) Test specification P30-8:

Prefix: X8 - Reach GA
Test case value: NOT X4 - Trigger event before-value
X4 - Trigger event
Expected Output: APPR

51) Test specification P30-9:

Prefix: X8 - Reach GA
Test case value: NOT X4 - Trigger event before-value
NOT X4 - Trigger event
Expected Output: GA

52) Test specification P31-1:

Prefix: X1 - Reach HDG
Test case value: NOT X8 - Trigger event before-value
X8 - Trigger event
Expected Output: GA

53) Test specification P31-2:

Prefix: X1 - Reach HDG
Test case value: X8 - Trigger event before-value
X8 - Trigger event
Expected Output: HDG

54) Test specification P31-3:

Prefix: X1 - Reach HDG
Test case value: NOT X8 - Trigger event before-value
NOT X8 - Trigger event
Expected Output: HDG

55) Test specification P31-4:

Prefix: X2 - Reach NAV
Test case value: NOT X8 - Trigger event before-value
X8 - Trigger event
Expected Output: GA

56) Test specification P31-5:
Prefix: X2 - Reach NAV
Test case value: X8 - Trigger event before-value
X8 - Trigger event
Expected Output: NAV

57) Test specification P31-6:
Prefix: X2 - Reach NAV
Test case value: NOT X8 - Trigger event before-value
NOT X8 - Trigger event
Expected Output: NAV

58) Test specification P31-7:
Prefix: X4 - Reach APPR
Test case value: NOT X8 - Trigger event before-value
X8 - Trigger event
Expected Output: GA

59) Test specification P31-8:
Prefix: X4 - Reach APPR
Test case value: X8 - Trigger event before-value
X8 - Trigger event
Expected Output: APPR

60) Test specification P31-9:
Prefix: X4 - Reach APPR
Test case value: NOT X8 - Trigger event before-value
NOT X8 - Trigger event
Expected Output: APPR

6.7.2 Transition Pair Coverage Level Requirements:
The pairs for the HDG Mode are:
P28 : (P20 or P29 or P30 or P31)
The pairs for the NAV Mode are:
P29 : (P21 or P22 or P28 or P30 or P31)
The pairs for the APPR Mode are:
P30 : (P23 or P24 or P28 or P29 or P31)
The pairs for the GA Mode are:

P31 : (P25 or P26 or P27 or P28 or P29 or P30)

<table>
<thead>
<tr>
<th>HDG</th>
<th>NAV</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
<th>X7</th>
<th>X8</th>
<th>HDG</th>
</tr>
</thead>
<tbody>
<tr>
<td>OR</td>
<td></td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>HDG</td>
</tr>
<tr>
<td>OR</td>
<td>APPR</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>HDG</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HDG</th>
<th>NAV</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
<th>X7</th>
<th>X8</th>
<th>NAV</th>
</tr>
</thead>
<tbody>
<tr>
<td>OR</td>
<td></td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>NAV</td>
</tr>
<tr>
<td>OR</td>
<td>APPR</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>NAV</td>
</tr>
<tr>
<td>OR</td>
<td>GA</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>NAV</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NAV</th>
<th>HDG</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
<th>X7</th>
<th>X8</th>
<th>HDG</th>
</tr>
</thead>
<tbody>
<tr>
<td>OR</td>
<td></td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>HDG</td>
</tr>
<tr>
<td>OR</td>
<td>NAV</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>HDG</td>
</tr>
<tr>
<td>OR</td>
<td>NAV</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>APPR</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>APPR</th>
<th>HDG</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
<th>X7</th>
<th>X8</th>
<th>APPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>OR</td>
<td></td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>APPR</td>
</tr>
<tr>
<td>OR</td>
<td>APPR</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>APPR</td>
</tr>
<tr>
<td>OR</td>
<td>APPR</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>APPR</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>APPR</th>
<th>HDG</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
<th>X7</th>
<th>X8</th>
<th>NAV</th>
</tr>
</thead>
<tbody>
<tr>
<td>OR</td>
<td></td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>NAV</td>
</tr>
<tr>
<td>OR</td>
<td>APPR</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>NAV</td>
</tr>
<tr>
<td>OR</td>
<td>APPR</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>GA</td>
</tr>
</tbody>
</table>
Test Specifications:

1) Test specification HDG-1:

Prefix:  
X2  - Reach NAV  
X4  - Reach APPR  
X8  - Reach GA  
Test case value:  
X1  - Trigger event P28  
X1  - Trigger event P20  
Expected Output: ROLL

2) Test specification HDG-2:

Prefix:  
X2  - Reach NAV  
X4  - Reach APPR  
X8  - Reach GA  
Test case value:  
X1  - Trigger event P28  
X2  - Trigger event P29  
Expected Output: NAV

3) Test specification HDG-3:

Prefix:  
X2  - Reach NAV  
X4  - Reach APPR  
X8  - Reach GA  
Test case value:  
X1  - Trigger event P28  
X4  - Trigger event P30  
Expected Output: APPR

4) Test specification HDG-4:

Prefix:  
X2  - Reach NAV  
X4  - Reach APPR
Test case value: X8 - Reach GA
Test case value: X1 - Trigger event P28
Test case value: X8 - Trigger event P31
Expected Output: GA

5) Test specification NAV-1:

Prefix:
X1 - Reach HDG
X4 - Reach APPR
X8 - Reach GA
Test case value: X2 - Trigger event P29
X2 - Trigger event P21
Expected Output: ROLL

6) Test specification NAV-2:

Prefix:
X1 - Reach HDG
X4 - Reach APPR
X8 - Reach GA
Test case value: X2 - Trigger event P29
X3 - Trigger event P22
Expected Output: ROLL

7) Test specification NAV-3:

Prefix:
X1 - Reach HDG
X4 - Reach APPR
X8 - Reach GA
Test case value: X2 - Trigger event P29
X1 - Trigger event P28
Expected Output: HDG

8) Test specification NAV-4:

Prefix:
X1 - Reach HDG
X4 - Reach APPR
X8 - Reach GA
Test case value: X2 - Trigger event P29
X4 - Trigger event P30
Expected Output: APPR

9) Test specification NAV-5:

Prefix:
X1 - Reach HDG
X4 - Reach APPR
X8 - Reach GA
Test case value: X2 - Trigger event P29
X8 - Trigger event P31
Expected Output: GA

10) Test specification APPR-1:
Prefix:  X1 - Reach HDG  
        X2 - Reach NAV  
        X8 - Reach GA  
Test case value:  X4 - Trigger event P30  
                               X4 - Trigger event P23  
Expected Output:  ROLL  

11) Test specification APPR-2:  
      Prefix:  X1 - Reach HDG  
                X2 - Reach NAV  
                X8 - Reach GA  
Test case value:  X4 - Trigger event P30  
                              X3 - Trigger event P24  
Expected Output:  ROLL  

12) Test specification APPR-3:  
      Prefix:  X1 - Reach HDG  
                X2 - Reach NAV  
                X8 - Reach GA  
Test case value:  X4 - Trigger event P30  
                              X1 - Trigger event P28  
Expected Output:  HDG  

13) Test specification APPR-4:  
      Prefix:  X1 - Reach HDG  
                X2 - Reach NAV  
                X8 - Reach GA  
Test case value:  X4 - Trigger event P30  
                              X2 - Trigger event P29  
Expected Output:  NAV  

14) Test specification APPR-5:  
      Prefix:  X1 - Reach HDG  
                X2 - Reach NAV  
                X8 - Reach GA  
Test case value:  X4 - Trigger event P30  
                              X8 - Trigger event P31  
Expected Output:  GA  

15) Test specification GA-1:  
      Prefix:  X1 - Reach HDG  
                X2 - Reach NAV  
                X4 - Reach APPR  
Test case value:  X8 - Trigger event P31  
                              X5 = True - Trigger event P25  
Expected Output:  ROLL
16) Test specification GA-2:

Prefix:  
X1 - Reach HDG  
X2 - Reach NAV  
X4 - Reach APPR  

Test case value:  
X8 - Trigger event P31  
X6 = True - Trigger event P26  

Expected Output: ROLL

17) Test specification GA-3:

Prefix:  
X1 - Reach HDG  
X2 - Reach NAV  
X4 - Reach APPR  

Test case value:  
X8 - Trigger event P31  
X7 = False - Trigger event P27  

Expected Output: ROLL

18) Test specification GA-4:

Prefix:  
X1 - Reach HDG  
X2 - Reach NAV  
X4 - Reach APPR  

Test case value:  
X8 - Trigger event P31  
X1 - Trigger event P28  

Expected Output: HDG

19) Test specification GA-5:

Prefix:  
X1 - Reach HDG  
X2 - Reach NAV  
X4 - Reach APPR  

Test case value:  
X8 - Trigger event P31  
X2 - Trigger event P29  

Expected Output: NAV

20) Test specification GA-6:

Prefix:  
X1 - Reach HDG  
X2 - Reach NAV  
X4 - Reach APPR  

Test case value:  
X8 - Trigger event P31  
X4 - Trigger event P30  

Expected Output: APPR
### 6.8 Active Lateral ROLL Submode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>Entry to ROLL mode</td>
<td>@T(mode_Active_Lateral = ROLL) WHEN (term_Roll_LE_Threshold OR mon_On_Ground)</td>
<td>Hdg_Hold</td>
</tr>
<tr>
<td>33</td>
<td>ROLL Hold</td>
<td>@T(term_SYNC AND term_Roll_LE_Threshold)</td>
<td>Hdg_Hold</td>
</tr>
<tr>
<td>34</td>
<td>ROLL Hold</td>
<td>@T(term_AP_Engaged) WHEN (term_Roll_LE_Threshold)</td>
<td>Hdg_Hold</td>
</tr>
<tr>
<td>35</td>
<td>ROLL Hold</td>
<td>@T(mon_On_Ground)</td>
<td>Hdg_Hold</td>
</tr>
<tr>
<td>36</td>
<td>Entry to ROLL mode</td>
<td>@T(mode_Active_Lateral = ROLL) WHEN (NOT mon_On_Ground AND NOT term_Roll_LE_Threshold)</td>
<td>ROLL Hold</td>
</tr>
<tr>
<td>37</td>
<td>Hdg Hold</td>
<td>@T(term_SYNC AND NOT term_Roll_LE_Threshold AND NOT mon_On_Ground)</td>
<td>ROLL Hold</td>
</tr>
<tr>
<td>38</td>
<td>Hdg Hold</td>
<td>@T(term_AP_Engaged) WHEN (NOT mon_On_Ground AND NOT term_Roll_LE_Threshold)</td>
<td>ROLL Hold</td>
</tr>
</tbody>
</table>

**Table 6.8: Active Lateral ROLL Submode Transition Table**
(Table A.8, pg. 75 in the FGS report)

**Definitions:**
- $X_1 = (\text{mode}\_\text{Active}\_\text{Lateral} = \text{ROLL})$
- $X_2 = \text{term}\_\text{Roll}\_\text{LE}\_\text{Threshold}$
- $X_3 = \text{mon}\_\text{On}\_\text{Ground}$
- $X_4 = \text{term}\_\text{SYNC}$
- $X_5 = \text{term}\_\text{AP}\_\text{Engaged}$
- $X_1' = \text{After-value of trigger event } X_1$
- $X_2' = \text{After-value of trigger event } X_2$
- $X_3' = \text{After-value of trigger event } X_3$
- $X_4' = \text{After-value of trigger event } X_4$
- $X_5' = \text{After-value of trigger event } X_5$

**6.8.1 Full predicate coverage level test case requirements:**

<table>
<thead>
<tr>
<th>Pre State</th>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$X_4$</th>
<th>$X_5$</th>
<th>$X_1'$</th>
<th>$X_2'$</th>
<th>$X_3'$</th>
<th>$X_4'$</th>
<th>$X_5'$</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P32 Entry to ROLL</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Hdg_Hold</td>
</tr>
<tr>
<td>Entry to ROLL</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Entry to ROLL</td>
</tr>
<tr>
<td>Entry to ROLL</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Entry to ROLL</td>
</tr>
<tr>
<td>Entry to ROLL</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Hdg_Hold</td>
</tr>
<tr>
<td>Entry to ROLL</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Entry to ROLL</td>
</tr>
<tr>
<td>Entry to ROLL</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Entry to ROLL</td>
</tr>
</tbody>
</table>

69
Test specifications:

1) Test specification P32-1:

Prefix: Entry to Roll
Test case value: X1 = False - Trigger before-value
                X2 = True - Condition variable
                X1 = True - Trigger event
Expected Output: Hdg_Hold

2) Test specification P32-2:

Prefix: Entry to Roll
Test case value: X1 = True - Trigger before-value
                X2 = True - Condition variable
                X1 = True - Trigger event
Expected Output: Entry to ROLL

3) Test specification P32-3:

Prefix: Entry to Roll
Test case value:  X1 = False - Trigger before-value
X2 = False - Condition variable
X1 = True - Trigger event

Expected Output: Entry to ROLL

4) Test specification P32-4:

Prefix: Entry to Roll
Test case value: X1 = False - Trigger before-value
X2 = True - Condition variable
X1 = False - Trigger event

Expected Output: Entry to ROLL

5) Test specification P32-5:

Prefix: Entry to Roll
Test case value: X1 = False - Trigger before-value
X3 = True - Condition variable
X1 = True - Trigger event

Expected Output: Hdg_Hold

6) Test specification P32-6:

Prefix: Entry to Roll
Test case value: X1 = True - Trigger before-value
X3 = True - Condition variable
X1 = True - Trigger event

Expected Output: Entry to ROLL

7) Test specification P32-7:

Prefix: Entry to Roll
Test case value: X1 = False - Trigger before-value
X3 = False - Condition variable
X1 = True - Trigger event

Expected Output: Entry to ROLL

8) Test specification P32-8:

Prefix: Entry to Roll
Test case value: X1 = False - Trigger before-value
X3 = True - Condition variable
X1 = False - Trigger event

Expected Output: Entry to ROLL

9) Test specification P33-1:

Prefix: X4 = True - Reach ROLL_Hold
X2 = False - Condition variable
X3 = False - Condition variable
X5 = True - Reach ROLL_Hold
Test case value: X2 = False - Trigger before-value
X4 = False - Trigger before-value
X2 = True - Trigger event
X4 = True - Trigger event
Expected Output: Hdg_Hold

10) Test specification P33-2:

Prefix: X4 = True - Reach ROLL_Hold
X2 = False - Condition variable
X3 = False - Condition variable
X5 = True - Reach ROLL_Hold
Test case value: X2 = True - Trigger before-value
X4 = True - Trigger before-value
X2 = True - Trigger event
X4 = True - Trigger event
Expected Output: ROLL_Hold

11) Test specification P33-3:

Prefix: X4 = True - Reach ROLL_Hold
X2 = False - Condition variable
X3 = False - Condition variable
X5 = True - Reach ROLL_Hold
Test case value: X2 = False - Trigger before-value
X4 = False - Trigger before-value
X2 = False - Trigger event
X4 = True - Trigger event
Expected Output: ROLL_Hold

12) Test specification P33-4:

Prefix: X4 = True - Reach ROLL_Hold
X2 = False - Condition variable
X3 = False - Condition variable
X5 = True - Reach ROLL_Hold
Test case value: X2 = True - Trigger before-value
X4 = True - Trigger before-value
X2 = True - Trigger event
X4 = False - Trigger event
Expected Output: ROLL_Hold

13) Test specification P34-1:

Prefix: X4 = True - Reach ROLL_Hold
X2 = False - Condition variable
X3 = False - Condition variable
Test case value: X5 = False - Trigger before-value
X2 = True - Condition variable
X5 = True - Trigger event
Expected Output: Hdg_Hold
14) Test specification P34-2:

Prefix:  
X4 = True - Reach ROLL_Hold  
X2 = False - Condition variable  
X3 = False - Condition variable  

Test case value:  
X5 = True - Trigger before-value  
X2 = True - Condition variable  
X5 = True - Trigger event  

Expected Output: ROLL_Hold

15) Test specification P34-3:

Prefix:  
X4 = True - Reach ROLL_Hold  
X2 = False - Condition variable  
X3 = False - Condition variable  

Test case value:  
X5 = False - Trigger before-value  
X2 = False - Condition variable  
X5 = True - Trigger event  

Expected Output: ROLL_Hold

16) Test specification P34-4:

Prefix:  
X4 = True - Reach ROLL_Hold  
X2 = False - Condition variable  
X3 = False - Condition variable  

Test case value:  
X5 = False - Trigger before-value  
X2 = True - Condition variable  
X5 = False - Trigger event  

Expected Output: ROLL_Hold

17) Test specification P35-1:

Prefix:  
X4 = True - Reach ROLL_Hold  
X2 = False - Condition variable  
X3 = False - Condition variable  

Test case value:  
X3 = False - Trigger before-value  
X3 = True - Trigger event  

Expected Output: Hdg_Hold

18) Test specification P35-2:

Prefix:  
X4 = True - Reach ROLL_Hold  
X2 = False - Condition variable  
X3 = False - Condition variable  

Test case value:  
X3 = True - Trigger before-value  
X3 = True - Trigger event  

Expected Output: ROLL_Hold

19) Test specification P35-3:

Prefix:  
X4 = True - Reach ROLL_Hold
X2 = False - Condition variable
X3 = False - Condition variable
Test case value: X3 = False - Trigger before-value
X2 = False - Trigger event
X3 = False - Trigger event
Expected Output: ROLL_Hold

20) Test specification P36-1:
Prefix: Entry to Roll
Test case value: X1 = False - Trigger before-value
X2 = False - Condition variable
X3 = False - Condition variable
X1 = True - Trigger event
Expected Output: ROLL_Hold

21) Test specification P36-2:
Prefix: Entry to Roll
Test case value: X1 = True - Trigger before-value
X2 = False - Condition variable
X3 = False - Condition variable
X1 = True - Trigger event
Expected Output: Entry to ROLL

22) Test specification P36-3:
Prefix: Entry to Roll
Test case value: X1 = False - Trigger before-value
X2 = True - Condition variable
X3 = False - Condition variable
X1 = False - Trigger event
Expected Output: Entry to ROLL

23) Test specification P36-4:
Prefix: Entry to Roll
Test case value: X1 = False - Trigger before-value
X2 = False - Condition variable
X3 = True - Condition variable
X1 = False - Trigger event
Expected Output: Entry to ROLL

24) Test specification P37-1:
Prefix: X1 = True - Reach Hdg_Hold
X5 = True
X3 = True
Test case value: X2 = True - Trigger before-value
X3 = True - Trigger before-value
X4 = False - Trigger before-value
X2 = False - Trigger event
X3 = False - Trigger event  
X4 = True - Trigger event  
Expected Output: ROLL_Hold

25) Test specification P37-2:

Prefix:  
X1 = True - Reach Hdg_Hold  
X5 = True  
X3 = True  
Test case value:  
X2 = False - Trigger before-value  
X3 = True - Trigger before-value  
X4 = False - Trigger before-value  
X2 = False - Trigger event  
X3 = False - Trigger event  
X4 = True - Trigger event  
Expected Output: Hdg_Hold

26) Test specification P37-3:

Prefix:  
X1 = True - Reach Hdg_Hold  
X5 = True  
X3 = True  
Test case value:  
X2 = True - Trigger before-value  
X3 = False - Trigger before-value  
X4 = False - Trigger before-value  
X2 = False - Trigger event  
X3 = False - Trigger event  
X4 = True - Trigger event  
Expected Output: Hdg_Hold

27) Test specification P37-4:

Prefix:  
X1 = True - Reach Hdg_Hold  
X5 = True  
X3 = True  
Test case value:  
X2 = True - Trigger before-value  
X3 = True - Trigger before-value  
X4 = True - Trigger before-value  
X2 = False - Trigger event  
X3 = False - Trigger event  
X4 = True - Trigger event  
Expected Output: Hdg_Hold

28) Test specification P37-5:

Prefix:  
X1 = True - Reach Hdg_Hold  
X5 = True  
X3 = True  
Test case value:  
X2 = True - Trigger before-value  
X3 = True - Trigger before-value  
X4 = False - Trigger before-value  

75
X2 = True - Trigger event
X3 = False - Trigger event
X4 = True - Trigger event

Expected Output: Hdg_Hold

29) Test specification P37-6:

Prefix:  X1 = True  - Reach Hdg_Hold
         X5 = True
         X3 = True

Test case value:  X2 = True - Trigger before-value
                  X3 = True - Trigger before-value
                  X4 = False - Trigger before-value
                  X2 = False - Trigger event
                  X3 = True - Trigger event
                  X4 = True - Trigger event

Expected Output: Hdg_Hold

30) Test specification P37-7:

Prefix:  X1 = True  - Reach Hdg_Hold
         X5 = True
         X3 = True

Test case value:  X2 = True - Trigger before-value
                  X3 = True - Trigger before-value
                  X4 = False - Trigger before-value
                  X2 = False - Trigger event
                  X3 = False - Trigger event
                  X4 = False - Trigger event

Expected Output: Hdg_Hold

31) Test specification P38-1:

Prefix:  X1 = True  - Reach Hdg_Hold
         X5 = True
         X3 = True

Test case value:  X5 = False - Trigger before-value
                  X2 = False - Condition variable
                  X3 = False - Condition variable
                  X5 = True - Trigger event

Expected Output: ROLL_Hold

32) Test specification P38-2:

Prefix:  X1 = True  - Reach Hdg_Hold
         X5 = True
         X3 = True

Test case value:  X5 = True - Trigger before-value
                  X2 = False - Condition variable
                  X3 = False - Condition variable
                  X5 = True - Trigger event
Expected Output: Hdg_Hold

33) Test specification P38-3:

Prefix: X1 = True - Reach Hdg_Hold  
X5 = True  
X3 = True  
Test case value: X5 = False - Trigger before-value  
X2 = True - Condition variable  
X3 = False - Condition variable  
X5 = True - Trigger event  
Expected Output: Hdg_Hold

34) Test specification P38-4:

Prefix: X1 = True - Reach Hdg_Hold  
X5 = True  
X3 = True  
Test case value: X5 = False - Trigger before-value  
X2 = False - Condition variable  
X3 = True - Condition variable  
X5 = True - Trigger event  
Expected Output: Hdg_Hold

35) Test specification P38-5:

Prefix: X1 = True - Reach Hdg_Hold  
X5 = True  
X3 = True  
Test case value: X5 = False - Trigger before-value  
X2 = False - Condition variable  
X3 = False - Condition variable  
X5 = False - Trigger event  
Expected Output: Hdg_Hold

6.8.2 Transition Pair Coverage Level Requirements:

The pairs for the ROLL_Hold Mode are:

(P37 or P38) : (P33 or P34 or P35)  
P36 : (P33 or P34 or P35)

The pairs for the Hdg_Hold Mode are:

(P33 or P34 or P35) : (P37 or P38) P32 : (P37 or P38)

<table>
<thead>
<tr>
<th>ROLL_Hold</th>
<th>1_Hdg_Hold</th>
<th>OR</th>
<th>Hdg_Hold</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>X2</td>
<td>X3</td>
<td>X4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hdg_Hold</td>
<td>-</td>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>
ROLL_Hold - T - T - Hdg_Hold
OR
ROLL_Hold - t - - T Hdg_Hold
OR
ROLL_Hold - - T - - Hdg_Hold

2. Entry to ROLL - f f T - ROLL_Hold
ROLL_Hold - T - T - Hdg_Hold
OR
ROLL_Hold - t - - T Hdg_Hold
OR
ROLL_Hold - - T - - Hdg_Hold

Hdg_Hold 1. ROLL_Hold - T - T - Hdg_Hold
OR
ROLL_Hold - t - - T Hdg_Hold
OR
ROLL_Hold - - T - - Hdg_Hold
Hdg_Hold - F F T - ROLL_Hold
OR
Hdg_Hold - f f - T ROLL_Hold

2. Entry to ROLL T t t - - Hdg_Hold
Hdg_Hold - F F T - ROLL_Hold
OR
Hdg_Hold - f f - T ROLL_Hold

Test specifications

1) Test specification ROLL_Hold-1:
Prefix: X2 = True
X4 = True - Reach Hdg_Hold
X5 = True
X2 = True - Reach Hdg_Hold
X3 = True - Reach Hdg_Hold
Test case value: X4 = True - P37 trigger event
X2 = False
X3 = False
X5 = True - P38 trigger event
X2 = False
X3 = False
X4 = True - P33 trigger event
X5 = True - P34 trigger event
X2 = True - Condition variable
X3 = True - P35 trigger event
Expected Output: Hdg_Hold

2) Test specification ROLL_Hold-2:
Prefix: - Reach Entry to ROLL
Test case value: X4 = True - P33 trigger event
X5 = True - P34 trigger event
X2 = True - Condition variable
X3 = True - P35 trigger event
Expected Output: Hdg_Hold

3) Test specification Hdg_Hold-1
Prefix: X4 = True - Reach ROLL_Hold
X2 = False
X3 = False
X5 = True - Reach ROLL_Hold
X2 = False
X3 = False
Test case value: X4 = True - P33 trigger event
X5 = True - P34 trigger event
X2 = True - Condition variable
X3 = True - P35 trigger event
X4 = True - P37 trigger event
X2 = False
X3 = False
X5 = True - P38 trigger event
X2 = False
X3 = False
Expected Output: ROLL_Hold

4) Test specification Hdg_Hold-2:
Prefix: - Reach Entry to ROLL
Test case value: X4 = True - P37 trigger event
X2 = False
X3 = False
X5 = True - P38 trigger event
X2 = False
X3 = False
Expected Output: ROLL_Hold

6.9 Active Lateral Navigation Submode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>39</td>
<td>Armed</td>
<td>@T(mode_Lateral_NAV_Track_Cond_Met AND Duration(INMODE) &gt; const_min_armed_period)</td>
<td>Track</td>
</tr>
</tbody>
</table>

Table 6.9: Active Lateral Navigation Submode Transition Table (Table A.9, pg. 75 in the FGS report)

Definitions:

- X1 = (term_Lateral_NAV_Track_Cond_Met)
- X2 = (Duration(INMODE) > const_min_armed_period)
- X1' = -- After-value of trigger event X1
- $X_2'$ -- After-value of trigger event $X_2$
- $\text{Pre-P39} = @(\text{NAV Switch Pressed})$

6.9.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_1'$</th>
<th>$X_2'$</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P39 Armed</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>Track</td>
</tr>
<tr>
<td>Armed</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>Armed</td>
</tr>
<tr>
<td>Armed</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>Armed</td>
</tr>
<tr>
<td>Armed</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>Armed</td>
</tr>
</tbody>
</table>

Test specifications:

1) Test specification P39-1:

- Prefix: Pre-P39
- Test case value: $X_1 = \text{False}$ - Trigger before-value
  $X_2 = \text{False}$ - Trigger before-value
  $X_1 = \text{True}$ - Trigger event
  $X_2 = \text{True}$ - Trigger event
- Expected Output: Track

2) Test specification P39-2:

- Prefix: Pre-P39
- Test case value: $X_1 = \text{True}$ - Trigger before-value
  $X_2 = \text{True}$ - Trigger before-value
  $X_1 = \text{True}$ - Trigger event
  $X_2 = \text{True}$ - Trigger event
- Expected Output: Armed

3) Test specification P39-3:

- Prefix: Pre-P39
- Test case value: $X_1 = \text{False}$ - Trigger before-value
  $X_2 = \text{False}$ - Trigger before-value
  $X_1 = \text{False}$ - Trigger event
  $X_2 = \text{True}$ - Trigger event
- Expected Output: Armed

4) Test specification P39-4:

- Prefix: Pre-P39
- Test case value: $X_1 = \text{False}$ - Trigger before-value
  $X_2 = \text{False}$ - Trigger before-value
  $X_1 = \text{True}$ - Trigger event
  $X_2 = \text{False}$ - Trigger event
- Expected Output: Armed
6.9.2 Transition Pair Coverage Level Requirements:

NONE.

6.10 Active Lateral Approach Submode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>Armed</td>
<td>$\neg T(\text{mode}<em>\text{Lateral}</em>\text{APPR}<em>\text{Track}</em>\text{Cond}<em>\text{Met} \text{AND} \text{Duration(INMODE)} &gt; \text{const}</em>\text{min}<em>\text{armed}</em>\text{period})$</td>
<td>Track</td>
</tr>
</tbody>
</table>

Table 6.10: Active Lateral Approach Submode Transition Table (Table A.10, pg. 75 in the FGS report)

Definitions:

- $X_1 = (\text{term}_\text{lateral}_\text{APPR}_\text{Track}_\text{Cond}_\text{Met})$
- $X_2 = (\text{Duration(INMODE)} > \text{const}_\text{min}_\text{armed}_\text{period})$
- $X_1' = \text{After-value of trigger event } X_1$
- $X_2' = \text{After-value of trigger event } X_2$
- Pre-P40 = $\neg (\text{APPR}_\text{Switch}_\text{Pressed})$

6.10.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>X1</th>
<th>X2</th>
<th>X1'</th>
<th>X2'</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P40</td>
<td>Armed</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>Armed</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>Armed</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>Armed</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td></td>
</tr>
</tbody>
</table>

Test specifications:

1) Test specification P40-1:

Prefix: Pre-P40 - Reach Armed

Test case value: $X_1 = \text{False}$ - Trigger before-value
$X_2 = \text{False}$ - Trigger before-value
$X_1 = \text{True}$ - Trigger event
$X_2 = \text{True}$ - Trigger event

Expected Output: Track

2) Test specification P40-2:

Prefix: Pre-P40 - Reach Armed

Test case value: $X_1 = \text{True}$ - Trigger before-value
$X_2 = \text{True}$ - Trigger before-value
$X_1 = \text{True}$ - Trigger event
$X_2 = \text{True}$ - Trigger event
Expected Output: Armed

3) Test specification P40-3:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Pre-P40</th>
<th>- Reach Armed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test case value:</td>
<td>X1 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X2 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X1 = False</td>
<td>- Trigger event</td>
</tr>
<tr>
<td></td>
<td>X2 = True</td>
<td>- Trigger event</td>
</tr>
</tbody>
</table>

Expected Output: Armed

4) Test specification P40-4:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Pre-P40</th>
<th>- Reach Armed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test case value:</td>
<td>X1 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X2 = False</td>
<td>- Trigger before-value</td>
</tr>
<tr>
<td></td>
<td>X1 = True</td>
<td>- Trigger event</td>
</tr>
<tr>
<td></td>
<td>X2 = False</td>
<td>- Trigger event</td>
</tr>
</tbody>
</table>

Expected Output: Armed

6.10.2 Transition Pair Coverage Level Requirements:

NONE.
### 6.11 Active Vertical Mode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>GA</td>
<td>@T(term_SYNC)</td>
<td>PITCH</td>
</tr>
<tr>
<td>42</td>
<td>VS OR APPR OR ALTSEL OR PITCH</td>
<td>@VS_Pitch_Wheel_Changed</td>
<td>PITCH</td>
</tr>
<tr>
<td>43</td>
<td>ALTSEL</td>
<td>@T(mode_Altitude_Select = ACTIVE)</td>
<td>ALTSEL</td>
</tr>
<tr>
<td>44</td>
<td>ALTSEL</td>
<td>@CHANGED(Preselected_Altitude) WHEN mode_Altitude_Select = ACTIVE/Capture</td>
<td>PITCH</td>
</tr>
<tr>
<td>45</td>
<td>ALTSEL</td>
<td>@CHANGED(Preselected_Altitude) WHEN mode_Altitude_Select = ACTIVE/Track</td>
<td>ALTHOLD</td>
</tr>
<tr>
<td>46</td>
<td>APPR OR ALTHOLD</td>
<td>@ALT_Switch_Pressed</td>
<td>ALTHOLD</td>
</tr>
<tr>
<td>47</td>
<td>ALTHOLD</td>
<td>@ALT_Switch_Pressed</td>
<td>PITCH</td>
</tr>
<tr>
<td>48</td>
<td>APPR OR VS</td>
<td>@VS_Switch_Pressed</td>
<td>VS</td>
</tr>
<tr>
<td>49</td>
<td>VS</td>
<td>@VS_Switch_Pressed</td>
<td>PITCH</td>
</tr>
<tr>
<td>50</td>
<td>APPR OR FLC</td>
<td>@FLC_Switch_Pressed</td>
<td>FLC</td>
</tr>
<tr>
<td>51</td>
<td>FLC</td>
<td>@FLC_Switch_Pressed</td>
<td>PITCH</td>
</tr>
<tr>
<td>52</td>
<td>ALTSEL OR ALTHOLD OR APPR OR FLC</td>
<td>CONTINUOUSLY WHEN term_Overspeed</td>
<td>FLC</td>
</tr>
<tr>
<td>53</td>
<td>APPR</td>
<td>@T(mode_Vertical_Appearance = TRACK)</td>
<td>APPR</td>
</tr>
<tr>
<td>54</td>
<td>APPR</td>
<td>@T(mode_Vertical_Appearance = TRACK) AND NOT @GA_Pressed</td>
<td>PITCH</td>
</tr>
<tr>
<td>55</td>
<td>GA</td>
<td>@GA_Pressed</td>
<td>GA</td>
</tr>
<tr>
<td>56</td>
<td>GA</td>
<td>@T(term_AP_Engaged)</td>
<td>PITCH</td>
</tr>
<tr>
<td>57</td>
<td>GA</td>
<td>@F(mode_Active_Lateral = GA)</td>
<td>PITCH</td>
</tr>
</tbody>
</table>

Table 6.11: Active Vertical Mode Transition Table

(Table A.11, pg. 81 in the FGS report)

**Definitions:**

- $X_1 = \text{term\_SYNC}$
- $X_2 = @\text{(VS\_Pitch\_Wheel\_Changed)}$
- $X_3 = (\text{mode\_Altitude\_Select} = \text{ACTIVE})$
- $X_4 = @\text{CHANGED(\text{preselected\_Altitude})}$
- $X_5 = (\text{mode\_Altitude\_Select} = \text{ACTIVE/Capture})$
- $X_6 = (\text{mode\_Altitude\_Select} = \text{ACTIVE/Track})$
- $X_7 = \text{term\_Overspeed}$
- $X_8 = (\text{mode\_Vertical\_Appearance} = \text{Track})$
- $X_9 = \text{term}_{\text{AP}} \_\text{Engaged}$
- $X_{10} = (\text{mode}_{\text{Active}} \_\text{Lateral} = \text{GA})$
- $X_{11} = @\text{ALT}_{\text{Switch}}\_\text{Pressed}$
- $X_{12} = @\text{VS}_{\text{Switch}}\_\text{Pressed}$
- $X_{13} = @\text{FLC}_{\text{Switch}}\_\text{Pressed}$
- $X_{14} = @\text{GA}\_\text{Pressed}$

**Full predicate coverage level test case requirements:**

<table>
<thead>
<tr>
<th>Pre</th>
<th>X1 X2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X13 X14</th>
<th>Post</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>X1' = T PITCH</td>
<td>X1' = T GA</td>
</tr>
<tr>
<td>P41 GA</td>
<td>F - - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X1' = T PITCH</td>
</tr>
<tr>
<td>GA</td>
<td>T - - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X1' = T GA</td>
</tr>
<tr>
<td>GA</td>
<td>T - - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X1' = T GA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pre</th>
<th>X1 X2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X13 X14</th>
<th>Post</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>X2' = PITCH</td>
<td>X2' = GA</td>
</tr>
<tr>
<td>P42 GA</td>
<td>-NOT@ - - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X2' = PITCH</td>
</tr>
<tr>
<td>GA</td>
<td>@ - - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X2' = GA</td>
</tr>
<tr>
<td>GA</td>
<td>-NOT@ - - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>NOT X2' GA</td>
</tr>
<tr>
<td>ALTHOLD</td>
<td>-NOT@ - - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X2' = ALTHOLD</td>
</tr>
<tr>
<td>ALTHOLD</td>
<td>@ - - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X2' = ALTHOLD</td>
</tr>
<tr>
<td>ALTHOLD</td>
<td>-NOT@ - - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>NOT X2' ALTHOLD</td>
</tr>
<tr>
<td>FLC</td>
<td>-NOT@ - - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X2' = PITCH</td>
</tr>
<tr>
<td>FLC</td>
<td>- @ - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X2' = FLC</td>
</tr>
<tr>
<td>FLC</td>
<td>-NOT@ - - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>NOT X2' FLC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pre</th>
<th>X1 X2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X13 X14</th>
<th>Post</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>X3' = T ALTHOLD</td>
<td>X3' = T GA</td>
</tr>
<tr>
<td>P43 GA</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T ALTHOLD</td>
</tr>
<tr>
<td>GA</td>
<td>- - T - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T GA</td>
</tr>
<tr>
<td>GA</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = F GA</td>
</tr>
<tr>
<td>VS</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T VS</td>
</tr>
<tr>
<td>VS</td>
<td>- - T - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = F VS</td>
</tr>
<tr>
<td>VS</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T VS</td>
</tr>
<tr>
<td>APPR</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T ALTHOLD</td>
</tr>
<tr>
<td>APPR</td>
<td>- - T - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T APPR</td>
</tr>
<tr>
<td>APPR</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = F APPR</td>
</tr>
<tr>
<td>PITCH</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T PITCH</td>
</tr>
<tr>
<td>PITCH</td>
<td>- - T - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T PITCH</td>
</tr>
<tr>
<td>PITCH</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = F PITCH</td>
</tr>
<tr>
<td>ALTHOLD</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T ALTHOLD</td>
</tr>
<tr>
<td>ALTHOLD</td>
<td>- - T - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T ALTHOLD</td>
</tr>
<tr>
<td>ALTHOLD</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T ALTHOLD</td>
</tr>
<tr>
<td>FLC</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T FLC</td>
</tr>
<tr>
<td>FLC</td>
<td>- - T - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = T FLC</td>
</tr>
<tr>
<td>FLC</td>
<td>- - F - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X3' = F FLC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pre</th>
<th>X1 X2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X13 X14</th>
<th>Post</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>X4' = PITCH</td>
<td>X4' = ALTHOLD</td>
</tr>
<tr>
<td>P44 ALTHOLD</td>
<td>- - @ t - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X4' = ALTHOLD</td>
</tr>
<tr>
<td>ALTHOLD</td>
<td>- @ t - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X4' = ALTHOLD</td>
</tr>
<tr>
<td>ALTHOLD</td>
<td>-NOT@ f - - - - - - - - - - - - - - - - - - - - - - -</td>
<td>X4' = ALTHOLD</td>
</tr>
</tbody>
</table>
ALTSEL - - NOT@ t - - - - - - - - - NOT X4' ALTSEL

P45 ALTSEL - - NOT@ t - - - - - - - - - X4' ALTHOLD
ALTSEL - - @ t - - - - - - - - - X4' ALTSEL
ALTSEL - - NOT@ f - - - - - - - - - X4' ALTSEL
ALTSEL - - NOT@ t - - - - - - - - - NOT X4' ALTSEL

P46 GA - - - - - - - - - NOT@ - - - X11' ALTHOLD
GA - - - - - - - - - @ - - - X11' GA
GA - - - - - - - - - NOT@ - - - NOT X11' GA
VS - - - - - - - - - NOT@ - - - X11' ALTHOLD
VS - - - - - - - - - @ - - - X11' VS
VS - - - - - - - - - NOT@ - - - NOT X11' VS
ALTSEL - - - - - - - - - NOT@ - - - X11' ALTHOLD
ALTSEL - - - - - - - - - @ - - - X11' ALTSEL
ALTSEL - - - - - - - - - NOT@ - - - NOT X11' ALTSEL
PITCH - - - - - - - - - NOT@ - - - X11' ALTHOLD
PITCH - - - - - - - - - @ - - - X11' PITCH
PITCH - - - - - - - - - NOT@ - - - NOT X11' PITCH
FLC - - - - - - - - - NOT@ - - - X11' ALTHOLD
FLC - - - - - - - - - @ - - - X11' FLC
FLC - - - - - - - - - NOT@ - - - NOT X11' FLC

P47 ALTHOLD - - - - - - - - - NOT@ - - - X11' PITCH
ALTHOLD - - - - - - - - - NOT@ - - - NOT X11' ALTHOLD

P48 GA - - - - - - - - - NOT@ - - - X12' VS
GA - - - - - - - - - @ - - - X12' GA
GA - - - - - - - - - NOT@ - - - NOT X12' GA
ALTSEL - - - - - - - - - NOT@ - - - X12' VS
ALTSEL - - - - - - - - - @ - - - X12' ALTSEL
ALTSEL - - - - - - - - - NOT@ - - - NOT X12' ALTSEL
PITCH - - - - - - - - - NOT@ - - - X12' PITCH
PITCH - - - - - - - - - @ - - - X12' ALTSEL
PITCH - - - - - - - - - NOT@ - - - NOT X12' ALTSEL
ALTHOLD - - - - - - - - - NOT@ - - - X12' ALTHOLD
ALTHOLD - - - - - - - - - NOT@ - - - NOT X12' ALTHOLD
FLC - - - - - - - - - NOT@ - - - X12' VS
FLC - - - - - - - - - @ - - - X12' FLC
FLC - - - - - - - - - NOT@ - - - NOT X12' FLC

P49 VS - - - - - - - - - NOT@ - - - X12' PITCH
VS - - - - - - - - - @ - - - X12' VS
VS - - - - - - - - - NOT@ - - - NOT X12' VS

P50 GA - - - - - - - - - NOT@ - - - X13' FLC
GA - - - - - - - - - @ - - - X13' GA
GA - - - - - - - - - NOT@ - - - NOT X13' GA
VS - - - - - - - - - NOT@ - - - X13' FLC
VS - - - - - - - - - - - - - - - @ - X13' VS
VS - - - - - - - - - - - - - - NOT@ - NOT X13' VS
ALTSEL - - - - - - - - - - - - @ NOT@ - X13' FLC
ALTSEL - - - - - - - - - - - - @ - X13' ALTSEL
ALTSEL - - - - - - - - - - - - @ NOT@ - NOT X13' ALTSEL
PITCH - - - - - - - - - - - - - - @ - X13' PITCH
PITCH - - - - - - - - - - - - - NOT@ - NOT X13' PITCH
ALTSEL - - - - - - - - - - - - @ - X13' ALTSEL
ALTSEL - - - - - - - - - - - - @ NOT@ - NOT X13' ALTSEL
PITCH - - - - - - - - - - - - - - @ - X13' FLC
PITCH - - - - - - - - - - - - - NOT@ - NOT X13' FLC
ALTHOLD - - - - - - - - - - - - @ - X13' ALTHOLD
ALTHOLD - - - - - - - - - - - - @ NOT@ - NOT X13' ALTHOLD

P51 FLC - - - - - - - - - - - - NOT@ - X13' PITCH
FLC - - - - - - - - - - - - @ - X13' FLC
FLC - - - - - - - - - - - - NOT@ - NOT X13' FLC

P52ALTSEL - - - - - - - t - - - - - - - - - - ALTSEL
ALTSEL - - - - - NOT@ t - t - - - - - - - - - - X4', FLC
ALTSEL - - - - - NOT@ t - f - - - - - - - - - - X4', PITCH
ALTSEL - - - - - NOT@ t - f - - - - - - - - - - X4', ALTSEL
ALTSEL - - - - - NOT@ t - t - - - - - - - - - - NOT X4', ALTSEL
ALTSEL - - - - - t - - - - - - NOT@ - - - - X12', FLC
ALTSEL - - - - - f - - - - - NOT@ - - - - X12', VS
ALTSEL - - - - - f - - - - - @ - - - - X12', ALTSEL
ALTSEL - - - - - t - - - - - NOT@ - - - - NOT X12', ALTSEL
ALTSEL - - - - - t - - - - - NOT@ - - - - X14', FLC
ALTSEL - - - - - f - - - - - NOT@ - - - - X14', GA
ALTSEL - - - - - t - - - - - @ - - - - X14', ALTSEL
ALTSEL - - - - - t - - - - - NOT@ NOT X14' ALTSEL

ALTHOLD - - - - - - - - - t - - - - - - - - - - ALTHOLD
ALTHOLD -NOT@ - - - - t - - - - - - - - - - X2', FLC
ALTHOLD -NOT@ - NOT@ - - - t - - - - - - - - - - X2', PITCH
ALTHOLD -@ - - - - t - - - - - - - - - - X2', ALTHOLD
ALTHOLD - NOT@ - - - t - - - - - - - NOT X2' ALTHOLD
ALTHOLD - NOT@ - - - t - - - - - - NOT@ - - - - X11', FLC
ALTHOLD - NOT@ - NOT@ - - - t - - - - NOT X11' ALTHOLD
ALTHOLD - NOT@ - NOT@ - - - t - - - - - NOT@ - - - - NOT X11' ALTHOLD
ALTHOLD - NOT@ - NOT@ - - - t - - - - - NOT@ - - - NOT X11' ALTHOLD
ALTHOLD - NOT@ - NOT@ - - - t - - - - - NOT@ - - NOT X11' ALTHOLD
ALTHOLD - NOT@ - NOT@ - - - t - - - - - @ - - NOT X11' ALTHOLD
ALTHOLD - NOT@ - NOT@ - - - t - - - - - @ NOT@ - NOT X11' ALTHOLD
ALTHOLD - NOT@ - NOT@ - - - t - - - - - @ NOT@ NOT X11' ALTHOLD

APPR - - - - - - - t - - - - - - - - - - APPR
APPR - - - - - - - t F - - - - - - - - @ X8' = True FLC

86
Test specifications:

1) Test specification P41-1:

Prefix: X14 - Reach GA
Test case value: X1 = False - Trigger event before-value
X1 = True - Trigger event
Expected Output: PITCH

2) Test specification P41-2:

Prefix: X14 - Reach GA
Test case value: \( X_1 = \text{True} \) - Trigger event before-value
\( X_1 = \text{True} \) - Trigger event
Expected Output: GA

3) Test specification P41-3:

Prefix: \( X_1 \)
Test case value: \( X_1 = \text{False} \) - Trigger event before-value
\( X_1 = \text{False} \) - Trigger event
Expected Output: GA

4) Test specification P42-1:

Prefix: \( X_1 \)
Test case value: \( \text{NOT} \ X_2 \) - Trigger event before-value
\( X_2 \) - Trigger event
Expected Output: PITCH

5) Test specification P42-2:

Prefix: \( X_1 \)
Test case value: \( X_2 \) - Trigger event before-value
\( X_2 \) - Trigger event
Expected Output: GA

6) Test specification P42-3:

Prefix: \( X_1 \)
Test case value: \( \text{NOT} \ X_2 \) - Trigger event before-value
\( \text{NOT} \ X_2 \) - Trigger event
Expected Output: GA

7) Test specification P42-4:

Prefix: \( X_4 = \text{True} \) - Reach ALTHOLD
\( X_5 = \text{True} \) - Condition variable
\( X_11 \) - Reach ALTHOLD
Test case value: \( \text{NOT} \ X_2 \) - Trigger event before-value
\( X_2 \) - Trigger event
Expected Output: PITCH

8) Test specification P42-5:

Prefix: \( X_4 = \text{True} \) - Reach ALTHOLD
\( X_5 = \text{True} \) - Condition variable
\( X_11 \) - Reach ALTHOLD
Test case value: \( X_2 \) - Trigger event before-value
\( X_2 \) - Trigger event
Expected Output: ALTHOLD

9) Test specification P42-6:
| Prefix: X14 | Reach ALTHOLD |
| Test case value: NOT X2 | Trigger event before-value |
| Test case value: NOT X2 | Trigger event |
| Expected Output: ALTHOLD |

10) Test specification P42-7:
- Reach FLC
| Prefix: X13 | Trigger event before-value |
| Test case value: NOT X2 | Trigger event |
| Expected Output: PITCH |

11) Test specification P42-8:
- Reach FLC
| Prefix: X13 | Trigger event before-value |
| Test case value: X2 | Trigger event |
| Expected Output: FLC |

12) Test specification P42-9:
| Prefix: X13 | Trigger event before-value |
| Test case value: NOT X2 | Trigger event |
| Test case value: NOT X2 | Trigger event |
| Expected Output: FLC |

13) Test specification P43-1:
- Reach GA
| Prefix: X14 | Trigger event before-value |
| Test case value: X3 = False | Trigger event |
| Test case value: X3 = True | Trigger event |
| Expected Output: ALTSEL |

14) Test specification P43-2:
| Prefix: X14 | Trigger event before-value |
| Test case value: X3 = True | Trigger event |
| Test case value: X3 = True | Trigger event |
| Expected Output: GA |

15) Test specification P43-3:
| Prefix: X14 | Trigger event before-value |
| Test case value: X3 = False | Trigger event |
| Test case value: X3 = False | Trigger event |
| Expected Output: GA |

16) Test specification P43-4:
| Prefix: X12 | Trigger event before-value |
| Test case value: X3 = False | Trigger event |
17) Test specification P43-5:

Prefix: X12
Test case value: X3 = True
X3 = True

Expected Output: ALTSEL

18) Test specification P43-6:

Prefix: X12
Test case value: X3 = False
X3 = False

Expected Output: VS

19) Test specification P43-7:

Prefix: X8 = True
Test case value: X3 = False
X3 = True

Expected Output: ALTSEL

20) Test specification P43-8:

Prefix: X8 = True
Test case value: X3 = True
X3 = True

Expected Output: APPR

21) Test specification P43-9:

Prefix: X8 = True
Test case value: X3 = False
X3 = False

Expected Output: APPR

22) Test specification P43-10:

Prefix: X2
Test case value: X3 = False
X3 = True

Expected Output: ALTSEL

23) Test specification P43-11:

Prefix: X2
Test case value: X3 = True
X3 = True

Expected Output: PITCH
24) Test specification P43-12:
Prefix: X2 - Reach PITCH
Test case value: X3 = False - Trigger event before-value
X3 = False - Trigger event
Expected Output: PITCH

25) Test specification P43-13:
Prefix: X4 - Reach ALTHOLD
X5 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: X3 = False - Trigger event before-value
X3 = True - Trigger event
Expected Output: ALTSEL

26) Test specification P43-14:
Prefix: X4 - Reach ALTHOLD
X5 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: X3 = True - Trigger event before-value
X3 = True - Trigger event
Expected Output: ALTHOLD

27) Test specification P43-15:
Prefix: X4 - Reach ALTHOLD
X5 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: X3 = False - Trigger event before-value
X3 = False - Trigger event
Expected Output: ALTHOLD

28) Test specification P43-16:
Prefix: X13 - Reach FLC
Test case value: X3 = False - Trigger event before-value
X3 = True - Trigger event
Expected Output: ALTHOLD

29) Test specification P43-17:
Prefix: X13 - Reach FLC
Test case value: X3 = True - Trigger event before-value
X3 = True - Trigger event
Expected Output: ALTHOLD

30) Test specification P43-18:
Prefix: X13 - Reach FLC
Test case value: $X_3 = False$ - Trigger event before-value

Expected Output: ALTHOLD

31) Test specification P44-1:

Prefix: $X_3 = True$ - Reach ALTSEL

Test case value: NOT $X_4$ - Trigger event before-value

$X_5 = True$ - Condition variable

$X_4$ - Trigger event

Expected Output: PITCH

32) Test specification P44-2:

Prefix: $X_3 = True$ - Reach ALTSEL

Test case value: $X_4$ - Trigger event before-value

$X_5 = True$ - Condition variable

$X_4$ - Trigger event

Expected Output: ALTSEL

33) Test specification P44-3:

Prefix: $X_3 = True$ - Reach ALTSEL

Test case value: NOT $X_4$ - Trigger event before-value

$X_5 = False$ - Condition variable

$X_4$ - Trigger event

Expected Output: ALTSEL

34) Test specification P44-4:

Prefix: $X_3 = True$ - Reach ALTSEL

Test case value: NOT $X_4$ - Trigger event before-value

$X_5 = True$ - Condition variable

NOT $X_4$ - Trigger event

Expected Output: ALTSEL

35) Test specification P45-1:

Prefix: $X_3 = True$ - Reach ALTSEL

Test case value: NOT $X_4$ - Trigger event before-value

$X_6 = True$ - Condition variable

$X_4$ - Trigger event

Expected Output: PITCH

36) Test specification P45-2:

Prefix: $X_3 = True$ - Reach ALTSEL

Test case value: $X_4$ - Trigger event before-value

$X_6 = True$ - Condition variable

$X_4$ - Trigger event

Expected Output: ALTSEL
37) Test specification P45-3:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X3 = True</td>
<td>NOT X4</td>
<td>ALTSEL</td>
</tr>
<tr>
<td>X5 = False</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Reach ALTSEL
- Trigger event before-value
- Condition variable
- Trigger event

38) Test specification P45-4:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X3 = True</td>
<td>NOT X4</td>
<td>ALTSEL</td>
</tr>
<tr>
<td>X5 = True</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOT X4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Reach ALTSEL
- Trigger event before-value
- Condition variable
- Trigger event

39) Test specification P46-1:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1 = True</td>
<td>NOT X11</td>
<td>ALTHOLD</td>
</tr>
<tr>
<td>X11</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Reach GA
- Trigger event before-value
- Trigger event

40) Test specification P46-2:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1 = True</td>
<td>X11</td>
<td>GA</td>
</tr>
<tr>
<td>X11</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Reach GA
- Trigger event before-value
- Trigger event

41) Test specification P46-3:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1 = True</td>
<td>NOT X11</td>
<td>GA</td>
</tr>
<tr>
<td>NOT X11</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Reach GA
- Trigger event before-value
- Trigger event

42) Test specification P46-4:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1 = True</td>
<td>NOT X11</td>
<td>ALTHOLD</td>
</tr>
<tr>
<td>NOT X11</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Reach VS
- Trigger event before-value
- Trigger event

43) Test specification P46-5:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1 = True</td>
<td>X11</td>
<td>VS</td>
</tr>
<tr>
<td>X11</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Reach VS
- Trigger event before-value
- Trigger event
44) Test specification P46-6:

Prefix: X12 - Reach VS
Test case value: NOT X11 - Trigger event before-value
NOT X11 - Trigger event
Expected Output: VS

45) Test specification P46-7:

Prefix: X3 = True - Reach ALTSEL
Test case value: NOT X11 - Trigger event before-value
X11 - Trigger event
Expected Output: ALTHOLD

46) Test specification P46-8:

Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - Trigger event before-value
X11 - Trigger event
Expected Output: ALTSEL

47) Test specification P46-9:

Prefix: X3 = True - Reach ALTSEL
Test case value: NOT X11 - Trigger event before-value
NOT X11 - Trigger event
Expected Output: ALTSEL

48) Test specification P46-10:

Prefix: X2 - Reach PITCH
Test case value: NOT X11 - Trigger event before-value
X11 - Trigger event
Expected Output: ALTHOLD

49) Test specification P46-11:

Prefix: X2 - Reach PITCH
Test case value: X11 - Trigger event before-value
X11 - Trigger event
Expected Output: PITCH

50) Test specification P46-12:

Prefix: X2 - Reach PITCH
Test case value: NOT X11 - Trigger event before-value
NOT X11 - Trigger event
Expected Output: PITCH

51) Test specification P46-13:
Prefix: X13 - Reach FLC
Test case value: NOT X11 - Trigger event before-value
X11 - Trigger event
Expected Output: ALTHOLD

52) Test specification P46-14:
Prefix: X13 - Reach FLC
Test case value: X11 - Trigger event before-value
X11 - Trigger event
Expected Output: FLC

53) Test specification P46-15:
Prefix: X13 - Reach FLC
Test case value: NOT X11 - Trigger event before-value
NOT X11 - Trigger event
Expected Output: FLC

54) Test specification P47-1:
Prefix: X4 = True - Reach ALTHOLD
X5 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: NOT X11 - Trigger event before-value
X11 - Trigger event
Expected Output: PITCH

55) Test specification P47-2:
Prefix: X4 = True - Reach ALTHOLD
X5 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: X11 - Trigger event before-value
X11 - Trigger event
Expected Output: ALTHOLD

56) Test specification P47-3:
Prefix: X4 = True - Reach ALTHOLD
X5 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: NOT X11 - Trigger event before-value
NOT X11 - Trigger event
Expected Output: ALTHOLD

57) Test specification P48-1:
Prefix: X14 - Reach GA
Test case value: NOT X12 - Trigger event before-value
X12 - Trigger event
Expected Output: VS

58) Test specification P48-2:

Prefix: X14                    - Reach GA
Test case value: X12          - Trigger event before-value
                      X12          - Trigger event
Expected Output: GA

59) Test specification P48-3:

Prefix: X14                    - Reach GA
Test case value: NOT X12      - Trigger event before-value
                      NOT X12      - Trigger event
Expected Output: GA

60) Test specification P48-4:

Prefix: X3 = True              - Reach ALTSEL
Test case value: NOT X12      - Trigger event before-value
                      X12          - Trigger event
Expected Output: VS

61) Test specification P48-5:

Prefix: X3 = True              - Reach ALTSEL
Test case value: X12          - Trigger event before-value
                      X12          - Trigger event
Expected Output: ALTSEL

62) Test specification P48-6:

Prefix: X3 = True              - Reach ALTSEL
Test case value: NOT X12      - Trigger event before-value
                      NOT X12      - Trigger event
Expected Output: ALTSEL

63) Test specification P48-7:

Prefix: X2                     - Reach PITCH
Test case value: NOT X12      - Trigger event before-value
                      X12          - Trigger event
Expected Output: VS

64) Test specification P48-8:

Prefix: X2                     - Reach PITCH
Test case value: X12          - Trigger event before-value
                      X12          - Trigger event
Expected Output: PITCH
65) Test specification P48-9:
Prefix: X2 - Reach PITCH
Test case value: NOT X12 - Trigger event before-value
NOT X12 - Trigger event
Expected Output: PITCH

66) Test specification P48-10:
Prefix: X4 - Reach ALTHOLD
X6 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: X12 - Trigger event before-value
NOT X12 - Trigger event
Expected Output: VS

67) Test specification P48-11:
Prefix: X4 - Reach ALTHOLD
X6 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: X12 - Trigger event before-value
X12 - Trigger event
Expected Output: ALTHOLD

68) Test specification P48-12:
Prefix: X4 - Reach ALTHOLD
X6 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: NOT X12 - Trigger event before-value
NOT X12 - Trigger event
Expected Output: ALTHOLD

69) Test specification P48-13:
Prefix: X13 - Reach FLC
Test case value: NOT X12 - Trigger event before-value
X12 - Trigger event
Expected Output: VS

70) Test specification P48-14:
Prefix: X13 - Reach FLC
Test case value: X12 - Trigger event before-value
X12 - Trigger event
Expected Output: FLC

71) Test specification P48-15:
Prefix: X13 - Reach FLC
Test case value: NOT X:2 - Trigger event before-value NOT X:2 - Trigger event
Expected Output: ALTHOLD

72) Test specification P49-1:
Prefix: X:12 - Reach VS
Test case value: NOT X:12 - Trigger event before-value X:12 - Trigger event
Expected Output: PITCH

73) Test specification P49-2:
Prefix: X:12 - Reach VS
Test case value: X:12 - Trigger event before-value X:12 - Trigger event
Expected Output: VS

74) Test specification P49-3:
Prefix: X:12 - Reach VS
Test case value: NOT X:12 - Trigger event before-value NOT X:12 - Trigger event
Expected Output: VS

75) Test specification P50-1:
Prefix: X:14 - Reach GA
Test case value: NOT X:13 - Trigger event before-value X:13 - Trigger event
Expected Output: FLC

76) Test specification P50-2:
Prefix: X:14 - Reach GA
Test case value: X:13 - Trigger event before-value X:13 - Trigger event
Expected Output: GA

77) Test specification P50-3:
Prefix: X:14 - Reach GA
Test case value: NOT X:13 - Trigger event before-value NOT X:13 - Trigger event
Expected Output: GA

78) Test specification P50-4:
Prefix: X:12 - Reach VS
Test case value: NOT X:13 - Trigger event before-value X:13 - Trigger event
Expected Output: FLC

79) Test specification P50-5:

| Prefix:   | X12 | - Reach VS |
| Test case value: | X13, X13 | - Trigger event before-value, Trigger event |
| Expected Output: | VS | |

80) Test specification P50-6:

| Prefix:   | X12 | - Reach VS |
| Test case value: | NOT X13, NOT X13 | - Trigger event before-value, Trigger event |
| Expected Output: | VS | |

81) Test specification P50-7:

| Prefix:   | X3 = True | - Reach ALTSEL |
| Test case value: | NOT X13, X13 | - Trigger event before-value, Trigger event |
| Expected Output: | FLC | |

82) Test specification P50-8:

| Prefix:   | X3 = True | - Reach ALTSEL |
| Test case value: | X13, X13 | - Trigger event before-value, Trigger event |
| Expected Output: | ALTSEL | |

83) Test specification P50-9:

| Prefix:   | X3 = True | - Reach ALTSEL |
| Test case value: | NOT X13, NOT X13 | - Trigger event before-value, Trigger event |
| Expected Output: | ALTSEL | |

84) Test specification P50-10:

| Prefix:   | X2 | - Reach PITCH |
| Test case value: | NOT X13, X13 | - Trigger event before-value, Trigger event |
| Expected Output: | FLC | |

85) Test specification P50-11:

| Prefix:   | X2 | - Reach PITCH |
| Test case value: | X13, X13 | - Trigger event before-value, Trigger event |
| Expected Output: | PITCH | |
86) Test specification P50-12:
Prefix: X2  - Reach PITCH
Test case value: NOT X13  - Trigger event before-value
          NOT X13  - Trigger event
Expected Output: PITCH

87) Test specification P50-13:
Prefix:  X4  - Reach ALTHOLD
          X6 = True  - Condition variable
          X11  - Reach ALTHOLD
Test case value: NOT X13  - Trigger event before-value
          X13  - Trigger event
Expected Output: FLC

88) Test specification P50-14:
Prefix:  X4  - Reach ALTHOLD
          X6 = True  - Condition variable
          X11  - Reach ALTHOLD
Test case value: X13  - Trigger event before-value
          X13  - Trigger event
Expected Output: ALTHOLD

89) Test specification P50-15:
Prefix:  X4  - Reach ALTHOLD
          X6 = True  - Condition variable
          X11  - Reach ALTHOLD
Test case value: NOT X13  - Trigger event before-value
          NOT X13  - Trigger event
Expected Output: ALTHOLD

90) Test specification P51-1:
Prefix: X13  - Reach FLC
Test case value: NOT X13  - Trigger event before-value
          X13  - Trigger event
Expected Output: PITCH

91) Test specification P51-2:
Prefix: X13  - Reach FLC
Test case value: X13  - Trigger event before-value
          X13  - Trigger event
Expected Output: FLC

92) Test specification P51-3:
Prefix: X13  - Reach FLC
93) Test specification P52-1:
Prefix: X3 = True - Reach ALTSEL
Test case value: X7 = True - Condition variable
Expected Output: FLC

94) Test specification P52-2:
Prefix: X3 = True - Reach ALTSEL
Test case value: NOT X4 - Trigger event before-value
X5 = True - Condition variable
X7 = True - Condition variable
X4 - Trigger event
Expected Output: FLC

95) Test specification P52-3:
Prefix: X3 = True - Reach ALTSEL
Test case value: NOT X4 - Trigger event before-value
X5 = True - Condition variable
X7 = False - Condition variable
X4 - Trigger event
Expected Output: PITCH

96) Test specification P52-4:
Prefix: X3 = True - Reach ALTSEL
Test case value: X4 - Trigger event before-value
X5 = True - Condition variable
X7 = True - Condition variable
X4 - Trigger event
Expected Output: ALTSEL

97) Test specification P52-5:
Prefix: X3 = True - Reach ALTSEL
Test case value: NOT X4 - Trigger event before-value
X5 = False - Condition variable
X7 = True - Condition variable
X4 - Trigger event
Expected Output: ALTSEL

98) Test specification P52-6:
Prefix: X3 = True - Reach ALTSEL
Test case value: NOT X4 - Trigger event before-value
X5 = True - Condition variable
X7 = True - Condition variable
NOT X4 - Trigger event

Expected Output: ALTSEL

99) Test specification P52-7:

Prefix: X3 = True - Reach ALTSEL
Test case value: NOT X12 - Trigger event before-value
X7 = True - Condition variable
X12 - Trigger event

Expected Output: FLC

100) Test specification P52-8:

Prefix: X3 = True - Reach ALTSEL
Test case value: NOT X12 - Trigger event before-value
X7 = False - Condition variable
X12 - Trigger event

Expected Output: VS

101) Test specification P52-9:

Prefix: X3 = True - Reach ALTSEL
Test case value: X12 - Trigger event before-value
X7 = True - Condition variable
X12 - Trigger event

Expected Output: ALTSEL

102) Test specification P52-10:

Prefix: X3 = True - Reach ALTSEL
Test case value: NOT X12 - Trigger event before-value
X7 = True - Condition variable
NOT X12 - Trigger event

Expected Output: ALTSEL

103) Test specification P52-11:

Prefix: X3 = True - Reach ALTSEL
Test case value: NOT X14 - Trigger event before-value
X7 = True - Condition variable
X14 - Trigger event

Expected Output: FLC

104) Test specification P52-12:

Prefix: X3 = True - Reach ALTSEL
Test case value: NOT X14 - Trigger event before-value
X7 = False - Condition variable
X14 - Trigger event

Expected Output: GA
105) Test specification P52-13:

Prefix: X3 = True
Test case value: X14 - Trigger event before-value
X7 = False - Condition variable
X14 - Trigger event
Expected Output: ALTSEL

106) Test specification P52-14:

Prefix: X3 = True
Test case value: NOT X14 - Trigger event before-value
X7 = False - Condition variable
NOT X14 - Trigger event
Expected Output: ALTSEL

107) Test specification P52-15:

Prefix: X11
X4 - Reach ALTHOLD
X6 = True - Condition variable
Test case value: X7 = True - Condition variable
Expected Output: ALTHOLD

108) Test specification P52-16:

Prefix: X11
X4 - Reach ALTHOLD
X6 = True - Condition variable
Test case value: NOT X2 - Trigger event before-value
X7 = True - Condition variable
X2 - Trigger event
Expected Output: FLC

109) Test specification P52-17:

Prefix: X11
X4 - Reach ALTHOLD
X6 = True - Condition variable
Test case value: NOT X2 - Trigger event before-value
X7 = False - Condition variable
X2 - Trigger event
Expected Output: PITCH

110) Test specification P52-18:

Prefix: X11
X4 - Reach ALTHOLD
X6 = True - Condition variable
Test case value: X2 - Trigger event before-value
X7 = True - Condition variable
X2 - Trigger event

Expected Output: ALTHOLD

111) Test specification P52-19:

Prefix: X11 - Reach ALTHOLD
        X4 - Reach ALTHOLD
        X6 = True - Condition variable

Test case value: NOT X2 - Trigger event before-value
                 X7 = True - Condition variable
                 NOT X2 - Trigger event

Expected Output: ALTHOLD

112) Test specification P52-20:

Prefix: X11 - Reach ALTHOLD
        X4 - Reach ALTHOLD
        X6 = True - Condition variable

Test case value: NOT X11 - Trigger event before-value
                 X7 = True - Condition variable
                 X11 - Trigger event

Expected Output: FLC

113) Test specification P52-21:

Prefix: X11 - Reach ALTHOLD
        X4 - Reach ALTHOLD
        X6 = True - Condition variable

Test case value: NOT X11 - Trigger event before-value
                 X7 = False - Condition variable
                 X11 - Trigger event

Expected Output: PITCH

114) Test specification P52-22:

Prefix: X11 - Reach ALTHOLD
        X4 - Reach ALTHOLD
        X6 = True - Condition variable

Test case value: X11 - Trigger event before-value
                 X7 = True - Condition variable
                 X11 - Trigger event

Expected Output: ALTHOLD

115) Test specification P52-23:

Prefix: X11 - Reach ALTHOLD
        X4 - Reach ALTHOLD
        X6 = True - Condition variable

Test case value: NOT X11 - Trigger event before-value
                 X7 = True - Condition variable
                 NOT X11 - Trigger event
Expected Output: ALTHOLD

116) Test specification P52-24:

Prefix:  
X11  - Reach ALTHOLD
X4   - Reach ALTHOLD
X6 = True - Condition variable
Test case value:  
NOT X12 - Trigger event before-value
X7 = True - Condition variable
X12 - Trigger event
Expected Output: FLC

117) Test specification P52-25:

Prefix:  
X11  - Reach ALTHOLD
X4   - Reach ALTHOLD
X6 = True - Condition variable
Test case value:  
NOT X12 - Trigger event before-value
X7 = False - Condition variable
X12 - Trigger event
Expected Output: VS

118) Test specification P52-26:

Prefix:  
X11  - Reach ALTHOLD
X4   - Reach ALTHOLD
X6 = True - Condition variable
Test case value:  
X12 - Trigger event before-value
X7 = True - Condition variable
X12 - Trigger event
Expected Output: ALTHOLD

119) Test specification P52-27:

Prefix:  
X11  - Reach ALTHOLD
X4   - Reach ALTHOLD
X6 = True - Condition variable
Test case value:  
NOT X12 - Trigger event before-value
X7 = True - Condition variable
NOT X12 - Trigger event
Expected Output: ALTHOLD

120) Test specification P52-28:

Prefix:  
X11  - Reach ALTHOLD
X4   - Reach ALTHOLD
X6 = True - Condition variable
Test case value:  
X14 - Trigger event before-value
X7 = True - Condition variable
NOT X14 - Trigger event
Expected Output: FLC
121) Test specification P52-29:

Prefix:
X11 - Reach ALTHOLD
X4 - Reach ALTHOLD
X6 = True - Condition variable

Test case value:
NOT X14 - Trigger event before-value
X7 = False - Condition variable
X14 - Trigger event

Expected Output: GA

122) Test specification P52-30:

Prefix:
X11 - Reach ALTHOLD
X4 - Reach ALTHOLD
X6 = True - Condition variable

Test case value:
X14 - Trigger event before-value
X7 = True - Condition variable
X14 - Trigger event

Expected Output: ALTHOLD

123) Test specification P52-31:

Prefix:
X11 - Reach ALTHOLD
X4 - Reach ALTHOLD
X6 = True - Condition variable

Test case value:
NOT X14 - Trigger event before-value
X7 = True - Condition variable
NOT X14 - Trigger event

Expected Output: ALTHOLD

124) Test specification P52-32:

Prefix:
X8 = True - Reach APPR

Test case value:
X7 = True - Condition variable

Expected Output: APPR

125) Test specification P52-33:

Prefix:
X8 = True - Reach APPR

Test case value:
X8 = False - Trigger event before-value
X14 - Trigger event before-value
X7 = True - Condition variable
X8 = True - Trigger event
NOT X14 - Trigger event

Expected Output: FLC

126) Test specification P52-34:

Prefix:
X8 = True - Reach APPR

Test case value:
X8 = False - Trigger event before-value
X14 - Trigger event before-value
X7 = False  - Condition variable
X8 = True   - Trigger event
NOT X14    - Trigger event

Expected Output: PITCH

127) Test specification P52-35:
Prefix: X8 = True   - Reach APPR
Test case value: X8 = True   - Trigger event before-value
X14          - Trigger event before-value
X7 = True    - Condition variable
X8 = True    - Trigger event
NOT X14     - Trigger event

Expected Output: APPR

128) Test specification P52-36:
Prefix: X8 = True   - Reach APPR
Test case value: X8 = False  - Trigger event before-value
NOT X14      - Trigger event before-value
X7 = True    - Condition variable
X8 = True    - Trigger event
NOT X14     - Trigger event

Expected Output: APPR

129) Test specification P52-37:
Prefix: X8 = True   - Reach APPR
Test case value: X8 = False  - Trigger event before-value
X14          - Trigger event before-value
X7 = True    - Condition variable
X8 = False  - Trigger event
NOT X14     - Trigger event

Expected Output: APPR

130) Test specification P52-38:
Prefix: X8 = True   - Reach APPR
Test case value: X8 = False  - Trigger event before-value
X14          - Trigger event before-value
X7 = True    - Condition variable
X8 = True    - Trigger event
X14         - Trigger event

Expected Output: APPR

131) Test specification P52-39:
Prefix: X8 = True   - Reach APPR
Test case value: NOT X14   - Trigger event before-value
X7 = True    - Condition variable
X14         - Trigger event

Expected Output: APPR
Expected Output: FLC

132) Test specification P52-40:
Prefix: \( X_8 = True \) - Reach APPR
Test case value: NOT \( X_{14} \) - Trigger event before-value
\( X_7 = False \) - Condition variable
\( X_{14} \) - Trigger event
Expected Output: APPR

133) Test specification P52-41:
Prefix: \( X_8 = True \) - Reach APPR
Test case value: \( X_{14} \) - Trigger event before-value
\( X_7 = True \) - Condition variable
\( X_{14} \) - Trigger event
Expected Output: APPR

134) Test specification P52-42:
Prefix: \( X_8 = True \) - Reach APPR
Test case value: NOT \( X_{14} \) - Trigger event before-value
\( X_7 = True \) - Condition variable
NOT \( X_{14} \) - Trigger event
Expected Output: APPR

135) Test specification P52-43:
Prefix: \( X_{13} \) - Reach FLC
Test case value: \( X_7 = True \) - Condition variable
Expected Output: FLC

136) Test specification P52-44:
Prefix: \( X_{13} \) - Reach FLC
Test case value: NOT \( X_2 \) - Trigger event before-value
\( X_7 = True \) - Condition variable
\( X_2 \) - Trigger event
Expected Output: FLC

137) Test specification P52-45:
Prefix: \( X_{13} \) - Reach FLC
Test case value: NOT \( X_2 \) - Trigger event before-value
\( X_7 = False \) - Condition variable
\( X_2 \) - Trigger event
Expected Output: PITCH

138) Test specification P52-46:
Prefix: \( X_{13} \) - Reach FLC
Test case value: $X_2$ - Trigger event before-value
$X_7 = True$ - Condition variable
$X_2$ - Trigger event
Expected Output: FLC

139) Test specification P52-47:

Prefix: $X_{13}$ - Reach FLC
Test case value: NOT $X_2$ - Trigger event before-value
$X_7 = True$ - Condition variable
$X_2$ - Trigger event
Expected Output: FLC

140) Test specification P52-48:

Prefix: $X_{13}$ - Reach FLC
Test case value: NOT $X_{12}$ - Trigger event before-value
$X_7 = True$ - Condition variable
$X_{12}$ - Trigger event
Expected Output: FLC

141) Test specification P52-49:

Prefix: $X_{13}$ - Reach FLC
Test case value: NOT $X_{12}$ - Trigger event before-value
$X_7 = False$ - Condition variable
$X_{12}$ - Trigger event
Expected Output: VS

142) Test specification P52-50:

Prefix: $X_{13}$ - Reach FLC
Test case value: $X_{12}$ - Trigger event before-value
$X_7 = True$ - Condition variable
$X_{12}$ - Trigger event
Expected Output: FLC

143) Test specification P52-51:

Prefix: $X_{13}$ - Reach FLC
Test case value: NOT $X_{12}$ - Trigger event before-value
$X_7 = True$ - Condition variable
NOT $X_{12}$ - Trigger event
Expected Output: FLC

144) Test specification P52-52:

Prefix: $X_{13}$ - Reach FLC
Test case value: NOT $X_{14}$ - Trigger event before-value
$X_7 = True$ - Condition variable
$X_{14}$ - Trigger event
Expected Output: FLC

145) Test specification P52-53:

Prefix: \text{X13} - Reach FLC
Test case value: \text{NOT X14} - Trigger event before-value
\text{X7 = False} - Condition variable
\text{X14} - Trigger event

Expected Output: GA

146) Test specification P52-54:

Prefix: \text{X13} - Reach FLC
Test case value: \text{X14} - Trigger event before-value
\text{X7 = True} - Condition variable
\text{X14} - Trigger event

Expected Output: FLC

147) Test specification P52-55:

Prefix: \text{X13} - Reach FLC
Test case value: \text{NOT X14} - Trigger event before-value
\text{X7 = True} - Condition variable
\text{NOT X14} - Trigger event

Expected Output: FLC

148) Test specification P52-56:

Prefix: \text{X13} - Reach FLC
Test case value: \text{NOT X13} - Trigger event before-value
\text{X7 = True} - Condition variable
\text{X13} - Trigger event

Expected Output: FLC

149) Test specification P52-57:

Prefix: \text{X13} - Reach FLC
Test case value: \text{NOT X13} - Trigger event before-value
\text{X7 = False} - Condition variable
\text{X13} - Trigger event

Expected Output: PITCH

150) Test specification P52-58:

Prefix: \text{X13} - Reach FLC
Test case value: \text{X13} - Trigger event before-value
\text{X7 = False} - Condition variable
\text{X13} - Trigger event

Expected Output: FLC

151) Test specification P52-59:
<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X13</td>
<td>NOT X13</td>
<td>FLC</td>
</tr>
<tr>
<td></td>
<td>X7 = False</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NOT X13</td>
<td>Trigger event</td>
</tr>
<tr>
<td></td>
<td></td>
<td>before-value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Condition variable</td>
</tr>
</tbody>
</table>

152) Test specification P53-1:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X14</td>
<td>X8 = False</td>
<td>APPR</td>
</tr>
<tr>
<td></td>
<td>X8 = True</td>
<td>Trigger event</td>
</tr>
</tbody>
</table>

153) Test specification P53-2:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X14</td>
<td>X8 = True</td>
<td>GA</td>
</tr>
<tr>
<td></td>
<td>X8 = True</td>
<td>Trigger event</td>
</tr>
</tbody>
</table>

154) Test specification P53-3:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X14</td>
<td>X8 = False</td>
<td>GA</td>
</tr>
<tr>
<td></td>
<td>X8 = False</td>
<td>Trigger event</td>
</tr>
</tbody>
</table>

155) Test specification P53-4:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X12</td>
<td>X8 = False</td>
<td>APPR</td>
</tr>
<tr>
<td></td>
<td>X8 = True</td>
<td>Trigger event</td>
</tr>
</tbody>
</table>

156) Test specification P53-5:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X12</td>
<td>X8 = True</td>
<td>VS</td>
</tr>
<tr>
<td></td>
<td>X8 = True</td>
<td>Trigger event</td>
</tr>
</tbody>
</table>

157) Test specification P53-6:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X12</td>
<td>X8 = False</td>
<td>VS</td>
</tr>
<tr>
<td></td>
<td>X8 = False</td>
<td>Trigger event</td>
</tr>
</tbody>
</table>

158) Test specification P53-7:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>X3 = True</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Reach ALTSEL</td>
</tr>
</tbody>
</table>
Test case value: \( X_8 = \text{False} \) - Trigger event before-value
\( X_8 = \text{True} \) - Trigger event
Expected Output: APPR

159) Test specification P53-8:
Prefix: \( X_3 = \text{True} \) - Reach ALTSEL
Test case value: \( X_8 = \text{True} \) - Trigger event before-value
\( X_8 = \text{True} \) - Trigger event
Expected Output: ALTSEL

160) Test specification P53-9:
Prefix: \( X_3 = \text{True} \) - Reach ALTSEL
Test case value: \( X_8 = \text{False} \) - Trigger event before-value
\( X_8 = \text{False} \) - Trigger event
Expected Output: ALTSEL

161) Test specification P53-10:
Prefix: \( X_2 \) - Reach PITCH
Test case value: \( X_8 = \text{False} \) - Trigger event before-value
\( X_8 = \text{True} \) - Trigger event
Expected Output: APPR

162) Test specification P53-11:
Prefix: \( X_2 \) - Reach PITCH
Test case value: \( X_8 = \text{False} \) - Trigger event before-value
\( X_8 = \text{True} \) - Trigger event
Expected Output: PITCH

163) Test specification P53-12:
Prefix: \( X_2 \) - Reach PITCH
Test case value: \( X_8 = \text{False} \) - Trigger event before-value
\( X_8 = \text{False} \) - Trigger event
Expected Output: PITCH

164) Test specification P53-13:
Prefix: \( X_4 \) - Reach ALTHOLD
\( X_6 = \text{False} \) - Condition variable
\( X_11 \) - Reach ALTHOLD
Test case value: \( X_8 = \text{False} \) - Trigger event before-value
\( X_8 = \text{False} \) - Trigger event
\( X_8 = \text{True} \) - Trigger event
Expected Output: APPR

165) Test specification P53-14:
Prefix: \( X_4 \) - Reach ALTHOLD
166) Test specification P53-15:
Prefix: X4
X6 = True
X11
Test case value: X8 = False
X8 = False
Expected Output: ALTHOLD

167) Test specification P53-16:
Prefix: X13
Test case value: X8 = False
X8 = True
Expected Output: APPR

168) Test specification P53-17:
Prefix: X13
Test case value: X8 = True
Expected Output: FLC

169) Test specification P53-18:
Prefix: X13
Test case value: X8 = False
Expected Output: FLC

170) Test specification P54-1:
Prefix: X8 = True
Test case value: X9 = False
X14
X9 = True
NOT X14
Expected Output: PITCH

171) Test specification P54-2:
Prefix: X8 = True
Test case value: X9 = True
X14
X9 = True
- Trigger event before-value
- Trigger event
NOT X14 - Trigger event

Expected Output: APPR

172) Test specification P54-3:

Prefix: X8 = True - Reach APPR
Test case value: X9 = False - Trigger event before-value
X9 = True - Trigger event
NOT X14 - Trigger event

Expected Output: APPR

173) Test specification P54-4:

Prefix: X8 = True - Reach APPR
Test case value: X9 = False - Trigger event before-value
X9 = True - Trigger event
NOT X14 - Trigger event

Expected Output: APPR

174) Test specification P54-5:

Prefix: X8 = True - Reach APPR
Test case value: X9 = False - Trigger event before-value
X9 = True - Trigger event
X14 - Trigger event

Expected Output: APPR

175) Test specification P55-1:

Prefix: X12 - Reach VS
Test case value: NOT X14 - Trigger event before-value
X14 - Trigger event

Expected Output: GA

176) Test specification P55-2:

Prefix: X12 - Reach VS
Test case value: X14 - Trigger event before-value
X14 - Trigger event

Expected Output: VS

177) Test specification P55-3:

Prefix: X12 - Reach VS
Test case value: NOT X14 - Trigger event before-value
NOT X14 - Trigger event

Expected Output: VS
178) Test specification P55-4:

Prefix: X8 - Reach APPR
Test case value: NOT Xi4 - Trigger event before-value
Xi4 - Trigger event
Expected Output: GA

179) Test specification P55-5:

Prefix: X8 - Reach APPR
Test case value: Xi4 - Trigger event before-value
Xi4 - Trigger event
Expected Output: APPR

180) Test specification P55-6:

Prefix: X8 - Reach APPR
Test case value: NOT Xi4 - Trigger event before-value
NOT Xi4 - Trigger event
Expected Output: APPR

181) Test specification P55-7:

Prefix: X3 = True - Reach ALTSEL
Test case value: NOT Xi4 - Trigger event before-value
Xi4 - Trigger event
Expected Output: GA

182) Test specification P55-8:

Prefix: X3 = True - Reach ALTSEL
Test case value: Xi4 - Trigger event before-value
Xi4 - Trigger event
Expected Output: ALTSEL

183) Test specification P55-9:

Prefix: X3 = True - Reach ALTSEL
Test case value: NOT Xi4 - Trigger event before-value
NOT Xi4 - Trigger event
Expected Output: ALTSEL

184) Test specification P55-10:

Prefix: X2 - Reach PITCH
Test case value: NOT Xi4 - Trigger event before-value
Xi4 - Trigger event
Expected Output: GA

185) Test specification P55-11:
Prefix: X2 - Reach PITCH
Test case value: X14 - Trigger event before-value
X14 - Trigger event
Expected Output: PITCH

186) Test specification P55-12:
Prefix: X2 - Reach PITCH
Test case value: NOT X14 - Trigger event before-value
NOT X14 - Trigger event
Expected Output: PITCH

187) Test specification P55-13:
Prefix: X4 - Reach ALTHOLD
X6 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: NOT X14 - Trigger event before-value
X14 - Trigger event
Expected Output: GA

188) Test specification P55-14:
Prefix: X4 - Reach ALTHOLD
X6 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: X14 - Trigger event before-value
X14 - Trigger event
Expected Output: ALTHOLD

189) Test specification P55-15:
Prefix: X4 - Reach ALTHOLD
X6 = True - Condition variable
X11 - Reach ALTHOLD
Test case value: NOT X14 - Trigger event before-value
NOT X14 - Trigger event
Expected Output: ALTHOLD

190) Test specification P55-16:
Prefix: X13 - Reach FLC
Test case value: NOT X14 - Trigger event before-value
X14 - Trigger event
Expected Output: GA

191) Test specification P55-17:
Prefix: X13 - Reach FLC
Test case value: X14 - Trigger event before-value
X14 - Trigger event
Expected Output: FLC

192) Test specification P55-18

Prefix: X13 - Reach FLC
Test case value: NOT X14 - Trigger event before-value
NOT X14 - Trigger event
Expected Output: FLC

193) Test specification P56-1

Prefix: X14 - Reach GA
Test case value: X9 = False - Trigger event before-value
X9 = True - Trigger event
Expected Output: PITCH

194) Test specification P56-2

Prefix: X14 - Reach GA
Test case value: X9 = True - Trigger event before-value
X9 = True - Trigger event
Expected Output: GA

195) Test specification P56-3

Prefix: X14 - Reach GA
Test case value: X9 = False - Trigger event before-value
X9 = False - Trigger event
Expected Output: GA

196) Test specification P57-1

Prefix: X14 - Reach GA
Test case value: X10 = True - Trigger event before-value
X10 = False - Trigger event
Expected Output: PITCH

197) Test specification P57-2

Prefix: X14 - Reach GA
Test case value: X10 = False - Trigger event before-value
X10 = False - Trigger event
Expected Output: GA

198) Test specification P56-3

Prefix: X14 - Reach GA
Test case value: X10 = True - Trigger event before-value
X10 = True - Trigger event
Expected Output: GA
Transition Pair Coverage Level Requirements:
The pairs for the PITCH Mode are:

(P41 or P42 or P44 or P47 or P49 or P51 or P54 or P56 or P57) :
(P43 or P46 or P48 or P50 or P52 or P53 or P55)

The pairs for the ALTSEL Mode are:

P43 : (P44 or P45 or P46 or P48 or P50 or P53 or P55)

The pairs for the ALTHOLD Mode are:

(P45 or P46) : (P42 or P43 or P47 or P48 or P50 or P53 or P55)

The pairs for VS Mode are:

P48 : (P43 or P46 or P49 or P50 or P52 or P53 or P55)

The pairs for FLC Mode are:

(P50 or P52) : (P42 or P43 or P46 or P48 or P51 or P53 or P55)

The pairs for APPR Mode are:

P53 : (P43 or P55 or P54)

The pairs for GA Mode are:

P55 : (P41 or P42 or P43 or P46 or P48 or P50 or P52 or P53)

<table>
<thead>
<tr>
<th>PITCH</th>
<th>OR</th>
<th>OR</th>
<th>OR</th>
<th>OR</th>
<th>OR</th>
<th>OR</th>
<th>OR</th>
<th>OR</th>
<th>OR</th>
<th>OR</th>
<th>OR</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>X1</td>
<td>X2</td>
<td>X3</td>
<td>X4</td>
<td>X5</td>
<td>X6</td>
<td>X7</td>
<td>X8</td>
<td>X9</td>
<td>X10</td>
<td>X11</td>
</tr>
<tr>
<td>OR</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>t</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
</tr>
<tr>
<td>VS</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
</tr>
<tr>
<td>OR</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
</tr>
<tr>
<td>FLC</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>APPR</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GA</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PITCH</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>PITCH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>PITCH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>-</td>
<td>VS</td>
</tr>
<tr>
<td>OR</td>
<td>PITCH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>PITCH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>PITCH</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ALTSEL</td>
<td>GA</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>VS</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>APPR</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>ALTHOLD</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>PITCH</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>FLC</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ALTSEL</td>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>t</td>
<td>t</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>t</td>
<td>f</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>t</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>-</td>
<td>t</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>t</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>f</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>t</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>f</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ALTHOLD</td>
<td>ALTSEL</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>@</td>
<td>t</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR</td>
<td>GA</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
OR
VS - - - - - - - - - - - - @ - - - ALTHOLD
OR
PITCH - - - - - - - - - - - - @ - - - ALTHOLD
OR
ALTSEL - - - - - - - - - - - - @ - - - ALTHOLD
OR
FLC - - - - - - - - - - - - @ - - - ALTHOLD
OR
ALTHOLD - @ - - - - t - - - - - - - FLC
OR
ALTHOLD - @ - - - - f - - - - - - - PITCH
OR
ALTHOLD - T - - - - - - - - - - - ALTS
OR
ALTHOLD - - - - - - - - - - - - @ - - - FLC
OR
ALTHOLD - - - - - - - - - - - - @ - - - PITCH
OR
ALTHOLD - - - - - - - - - - - - @ - - - FLC
OR
ALTHOLD - - - - - - - - - - - - @ - - - APPR
OR
ALTHOLD - - - - - - - - - - - - @ - - - GA
OR
VS - - - - - - - - - - - - @ - - - VS
OR
ALTSEL - - - - - - - - - - - - @ - - - VS
OR
ALTHOLD - - - - - - - - - - - - @ - - - VS
OR
PITCH - - - - - - - - - - - - @ - - - VS
OR
FLC - - - - - - - - - - - - @ - - - VS
OR
VS - - T - - - - - - - - - - - ALTS
OR
VS - - - - - - - - - - - - @ - - - ALTHOLD
OR
VS - - - - - - - - - - - - @ - - - PITCH
OR
VS - - - - - - - - - - - - @ - - - FLC
OR
VS - - - - - - - - - - - - T - - - - - APPR
VS  - - - - - - - - - - - - @ GA
FLC
GA  - - - - - - - - - - - - - O - FLC
OR
VS  - - - - - - - - - - - - - O - FLC
OR
ALTSEL - - - - - - - - - - - - - O - FLC
OR
ALTSEL - - - - - - - - - - - - @ - FLC
OR
ALTSEL - - - - - - - - - - - - - O - FLC
OR
ALTSEL - - - - - - - - - - - - - O - FLC
OR
ALTHOLD - - - - - - - - - - - - - O - FLC
OR
ALTHOLD - - - - - - - - - - - - - O - FLC
OR
ALTHOLD - - - - - - - - - - - - - O - FLC
OR
PITCH - - - - - - - - - - - - - O - FLC
OR
APPR - - - - - - - - - - - - - NOT@ FLC
OR
APPR - - - - - - - - - - - - - @ FLC
FLC  - - - - - - - - - - - - - PITCH
OR
FLC  - - - - - - - - - - - - - ALTSEL
OR
FLC  - - - - - - - - - - - - - @ - FLC
OR
FLC  - - - - - - - - - - - - - @ - FLC
OR
FLC  - - - - - - - - - - - - - @ - FLC
OR
FLC  - - - - - - - - - - - - - @ - FLC
OR
APPR  - - - - - - - - - - - - - APPR
OR
APPR  - - - - - - - - - - - - - GA
APPR  - - - - - - - - - - - - - GA
OR
VS  - - - - - - - - - - - - - APPR
OR
ALTSEL  - - - - - - - - - - - - - APPR
OR
ALTHOLD  - - - - - - - - - - - - - APPR
APPR
OR
PITCH - - - - - - - T - - - - - - APPR
OR
FLC - - - - - - - T - - - - - - APPR
APRR - - T - - - - - - - - - - ALTSEL
OR
APRR - - - - - - - t T - - - - - NOT@ FLC
OR
APRR - - - - - - - f T - - - - - NOT@ PITCH
OR
APRR - - - - - - - t - - - - - @ FLC
OR
APRR - - - - - - - f - - - - - @ GA

GA VS - - - - - - - - - - - - @ GA
OR
ALTSEL - - - - - - - - - - - - @ GA
OR
ALTHOLD - - - - - - - - - - - - @ GA
OR
PITCH - - - - - - - - - - - - @ GA
OR
APPR - - - - - - - - - - - - @ GA
OR
FLC - - - - - - - - - - - - @ GA

GA T - - - - - - - - - - - - @ PITCH
OR
GA - @ - - - - - - - - - - - - @ PITCH
OR
GA - - T - - - - - - - - - - - - ALTSEL
OR
GA - - - - - - - - - - - - @ - - - ALTHOLD
OR
GA - - - - - - - - - - - - @ - - - VS
OR
GA - - - - - - - - - - - - @ - - FLC
OR
GA - - - - - - - - - - - - T - - - - - APPR
OR
GA - - - - - - - - - - - - T - - - - - PITCH
OR
GA - - - - - - - - - - F - - - - - PITCH

Test Specifications:

1) Test specification PITCH-GA-1

Prefix: X14 - Reach GA
Test case value: X1 - P41 trigger event
X3 = True - P43 trigger event

Expected Output: ALTSEL

2) Test specification PITCH-GA-2

Prefix: X14 - Reach GA
Test case value: X1 - P41 trigger event
X11 - P46 trigger event

Expected Output: ALTHOLD

3) Test specification PITCH-GA-3

Prefix: X14 - Reach GA
Test case value: X1 - P41 trigger event
X12 - P48 trigger event

Expected Output: VS

4) Test specification PITCH-GA-4

Prefix: X14 - Reach GA
Test case value: X1 - P41 trigger event
X13 - P50 trigger event

Expected Output: FLC

5) Test specification PITCH-GA-5

Prefix: X14 - Reach GA
Test case value: X1 - P41 trigger event
X8 = True - P53 trigger event

Expected Output: APPR

6) Test specification PITCH-GA-6

Prefix: X14 - Reach GA
Test case value: X1 - P41 trigger event
X14 - P55 trigger event

Expected Output: GA

7) Test specification PITCH-GA-7

Prefix: X14 - Reach GA
Test case value: X2 - P42 trigger event
X3 = True - P43 trigger event

Expected Output: ALTSEL

8) Test specification PITCH-GA-8

Prefix: X14 - Reach GA
Test case value: X2 - P42 trigger event
X11 - P46 trigger event

Expected Output: ALTHOLD
9) Test specification PITCH-GA-9
Prefix: X14 - Reach GA
Test case value: X2 - P42 trigger event
X12 - P48 trigger event
Expected Output: VS

10) Test specification PITCH-GA-10
Prefix: X14 - Reach GA
Test case value: X2 - P42 trigger event
X12 - P50 trigger event

11) Test specification PITCH-GA-11
Prefix: X14 - Reach GA
Test case value: X2 - P42 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

12) Test specification PITCH-GA-12
Prefix: X14 - Reach GA
Test case value: X2 - P42 trigger event
X12 - P55 trigger event
Expected Output: GA

13) Test specification PITCH-GA-13
Prefix: X14 - Reach GA
Test case value: X9 = True - P56 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

14) Test specification PITCH-GA-14
Prefix: X14 - Reach GA
Test case value: X9 = True - P56 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

15) Test specification PITCH-GA-15
Prefix: X14 - Reach GA
Test case value: X9 = True - P56 trigger event
X12 - P48 trigger event
Expected Output: VS

16) Test specification PITCH-GA-16
Prefix: X14 - Reach GA
Test case value: X9 = True - P56 trigger event
X13 - P50 trigger event
Expected Output: FLC

17) Test specification PITCH-GA-17

Prefix: X14 - Reach GA
Test case value: X9 = True - P56 trigger event
X8 = True - P53 trigger event
Expected Output: FLC

18) Test specification PITCH-GA-18

Prefix: X14 - Reach GA
Test case value: X9 = True - P56 trigger event
X14 - P55 trigger event
Expected Output: GA

19) Test specification PITCH-GA-19

Prefix: X14 - Reach GA
Test case value: X10 = False - P57 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

20) Test specification PITCH-GA-20

Prefix: X14 - Reach GA
Test case value: X10 = False - P57 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

21) Test specification PITCH-GA-21

Prefix: X14 - Reach GA
Test case value: X10 = False - P57 trigger event
X12 - P48 trigger event
Expected Output: VS

22) Test specification PITCH-GA-22

Prefix: X14 - Reach GA
Test case value: X10 = False - P57 trigger event
X13 - P50 trigger event
Expected Output: FLC

23) Test specification PITCH-GA-23

Prefix: X14 - Reach GA
Test case value: X10 = False - P57 trigger event
24) Test specification PITCH-GA-24

Prefix: X14 - Reach GA
Test case value: X10 = False - P57 trigger event
X8 = True - P53 trigger event
Expected Output: GA

25) Test specification PITCH-ALTHOLD-1

Prefix: X11 - Reach ALTHOLD
Test case value: X2 - P42 trigger event
X3 = True - P43 trigger event
Expected Output: ALTHOLD

26) Test specification PITCH-ALTHOLD-2

Prefix: X11 - Reach ALTHOLD
Test case value: X2 - P42 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

27) Test specification PITCH-ALTHOLD-3

Prefix: X11 - Reach ALTHOLD
Test case value: X2 - P42 trigger event
X12 - P48 trigger event
Expected Output: VS

28) Test specification PITCH-ALTHOLD-4

Prefix: X11 - Reach ALTHOLD
Test case value: X2 - P42 trigger event
X13 - P50 trigger event
Expected Output: FLC

29) Test specification PITCH-ALTHOLD-5

Prefix: X11 - Reach ALTHOLD
Test case value: X2 - P42 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

30) Test specification PITCH-ALTHOLD-6

Prefix: X11 - Reach ALTHOLD
Test case value: X2 - P42 trigger event
X14 - P55 trigger event
Expected Output: GA
31) Test specification PITCH-ALTHOLD-7

Prefix:  X11  - Reach ALTHOLD
Test case value:  X11  - P47 trigger event
               X3 = True  - P43 trigger event
Expected Output:  ALTSEL

32) Test specification PITCH-ALTHOLD-8

Prefix:  X11  - Reach ALTHOLD
Test case value:  X11  - P47 trigger event
               X11  - P46 trigger event
Expected Output:  ALTHOLD

33) Test specification PITCH-ALTHOLD-9

Prefix:  X11  - Reach ALTHOLD
Test case value:  X11  - P47 trigger event
               X12  - P48 trigger event
Expected Output:  VS

34) Test specification PITCH-ALTHOLD-10

Prefix:  X11  - Reach ALTHOLD
Test case value:  X11  - P47 trigger event
               X11  - P50 trigger event
Expected Output:  FLC

35) Test specification PITCH-ALTHOLD-11

Prefix:  X11  - Reach ALTHOLD
Test case value:  X11  - P47 trigger event
               X8 = True  - P53 trigger event
Expected Output:  APPR

36) Test specification PITCH-ALTHOLD-12

Prefix:  X11  - Reach ALTHOLD
Test case value:  X2  - P42 trigger event
               X14  - P55 trigger event
Expected Output:  GA

37) Test specification PITCH-FLC-1

Prefix:  X13  - Reach FLC
Test case value:  X2  - P42 trigger event
               X3 = True  - P43 trigger event
Expected Output:  ALTSEL

38) Test specification PITCH-FLC-2
Prefix: X13
Test case value: X2
- P42 trigger event
X11
- P46 trigger event
Expected Output: ALTHOLD

39) Test specification PITCH-FLC-3
Prefix: X13
Test case value: X2
- P42 trigger event
X12
- P48 trigger event
Expected Output: VS

40) Test specification PITCH-FLC-4
Prefix: X13
Test case value: X2
- P42 trigger event
X13
- P50 trigger event
Expected Output: FLC

41) Test specification PITCH-FLC-5
Prefix: X13
Test case value: X2
- P42 trigger event
X8 = True
- P53 trigger event
Expected Output: APPR

42) Test specification PITCH-FLC-6
Prefix: X13
Test case value: X2
- P42 trigger event
X14
- P55 trigger event
Expected Output: GA

43) Test specification PITCH-FLC-7
Prefix: X13
Test case value: X13
- P51 trigger event
X3 = True
- P43 trigger event
Expected Output: ALTSEL

44) Test specification PITCH-FLC-8
Prefix: X13
Test case value: X13
- P51 trigger event
X11
- P46 trigger event
Expected Output: ALTHOLD

45) Test specification PITCH-FLC-9
Prefix: X13
Test case value: X13
- P51 trigger event
46) Test specification PITCH-FLC-10

Prefix: X13 - Reach FLC
Test case value: X13 - P51 trigger event
X13 - P50 trigger event
Expected Output: FLC

47) Test specification PITCH-FLC-11

Prefix: X13 - Reach FLC
Test case value: X13 - P51 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

48) Test specification PITCH-FLC-12

Prefix: X13 - Reach FLC
Test case value: X13 - P51 trigger event
X14 - P55 trigger event
Expected Output: GA

49) Test specification PITCH-ALTSEL-1

Prefix: X3 = True - Reach ALTSEL
Test case value: X4 - P44 trigger event
X5 = True - Condition variable
X3 = True - P43 trigger event
Expected Output: ALTSEL

50) Test specification PITCH-ALTSEL-2

Prefix: X3 = True - Reach ALTSEL
Test case value: X4 - P44 trigger event
X5 = True - Condition variable
X11 - P46 trigger event
Expected Output: ALTHOLD

51) Test specification PITCH-ALTSEL-3

Prefix: X3 = True - Reach ALTSEL
Test case value: X4 - P44 trigger event
X5 = True - Condition variable
X12 - P48 trigger event
Expected Output: VS

52) Test specification PITCH-ALTSEL-4

Prefix: X3 = True - Reach ALTSEL
Test case value: X₄ = True - Condition variable
X₅ = True - P₄₄ trigger event
X₁₃ = True - P₅₀ trigger event
Expected Output: FLC

53) Test specification PITCH-ALTSEL-5

Prefix: X₃ = True - Reach ALTSEL
Test case value: X₄ = True - P₄₄ trigger event
X₅ = True - Condition variable
X₈ = True - P₅₃ trigger event
Expected Output: APPR

54) Test specification PITCH-ALTSEL-6

Prefix: X₃ = True - Reach ALTSEL
Test case value: X₄ = True - P₄₄ trigger event
X₅ = True - Condition variable
X₁₄ = True - P₅₃ trigger event
Expected Output: GA

55) Test specification PITCH-APPR-1

Prefix: X₃ = True - Reach APPR
Test case value: X₈ = True
NOT X₁₄ - P₅₄ trigger event
X₃ = True - P₄₃ trigger event
Expected Output: ALTSEL

56) Test specification PITCH-APPR-2

Prefix: X₃ = True - Reach APPR
Test case value: X₈ = True
NOT X₁₄ - P₅₄ trigger event
X₁₁ = True - P₄₆ trigger event
Expected Output: ALTHOLD

57) Test specification PITCH-APPR-3

Prefix: X₃ = True - Reach APPR
Test case value: X₈ = True
NOT X₁₄ - P₅₄ trigger event
X₁₂ = True - P₄₈ trigger event
Expected Output: VS

58) Test specification PITCH-APPR-4

Prefix: X₃ = True - Reach APPR
Test case value: X₈ = True
NOT X₁₄ - P₅₄ trigger event
X₁₃ = True - P₅₀ trigger event
Expected Output: FLC

59) Test specification PITCH-APPR-5

Prefix: X3 = True - Reach APPR
Test case value: X8 = True
NOT X14 - P54 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

60) Test specification PITCH-APPR-6

Prefix: X3 = True - Reach APPR
Test case value: X8 = True
NOT X14 - P54 trigger event
X14 - P55 trigger event
Expected Output: GA

61) Test specification PITCH-VS-1

Prefix: X12 - Reach VS
Test case value: X12 - P54 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSELT

62) Test specification PITCH-VS-2

Prefix: X12 - Reach VS
Test case value: X12 - P54 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

63) Test specification PITCH-VS-3

Prefix: X12 - Reach VS
Test case value: X12 - P54 trigger event
X12 - P48 trigger event
Expected Output: VS

64) Test specification PITCH-VS-4

Prefix: X12 - Reach VS
Test case value: X12 - P54 trigger event
X13 - P50 trigger event
Expected Output: FLC

65) Test specification PITCH-VS-5

Prefix: X12 - Reach VS
Test case value: X12 - P54 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

66) Test specification PITCH-VS-6

Prefix: X:12 - Reach VS
Test case value: X:12 - P54 trigger event
X:14 - P55 trigger event
Expected Output: GA

67) Test specification ALTSEL-GA-1

Prefix: X:14 - Reach GA
Test case value: X:3 = True - P43 trigger event
X:4 - P44 trigger event
X:5 = True - Condition variable
X:7 = True - Condition variable
Expected Output: FLC

68) Test specification ALTSEL-GA-2

Prefix: X:14 - Reach GA
Test case value: X:3 = True - P43 trigger event
X:4 - P44 trigger event
X:5 = True - Condition variable
X:7 = False - Condition variable
Expected Output: PITCH

69) Test specification ALTSEL-GA-3

Prefix: X:14 - Reach GA
Test case value: X:3 = True - P43 trigger event
X:4 - P45 trigger event
X:6 = True - Condition variable
Expected Output: ALTHOLD

70) Test specification ALTSEL-GA-4

Prefix: X:14 - Reach GA
Test case value: X:3 = True - P43 trigger event
X:11 - P46 trigger event
Expected Output: ALTHOLD

71) Test specification ALTSEL-GA-5

Prefix: X:14 - Reach GA
Test case value: X:3 = True - P43 trigger event
X:12 - P52 trigger event
X:7 = True - Condition variable
Expected Output: FLC

72) Test specification ALTSEL-GA-6
Prefix: X14 - Reach GA
Test case value: X3 = True - P43 trigger event
X12 - P48 trigger event
X7 = False - Condition variable
Expected Output: VS

73) Test specification ALTSEL-GA-7
Prefix: X14 - Reach GA
Test case value: X3 = True - P43 trigger event
X13 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

74) Test specification ALTSEL-GA-8
Prefix: X14 - Reach GA
Test case value: X3 = True - P43 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

75) Test specification ALTSEL-GA-9
Prefix: X14 - Reach GA
Test case value: X3 = True - P43 trigger event
X14 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

76) Test specification ALTSEL-GA-10
Prefix: X14 - Reach GA
Test case value: X3 = True - P43 trigger event
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

77) Test specification ALTSEL-VS-1
Prefix: X12 - Reach VS
Test case value: X3 = True - P43 trigger event
X4 - P52 trigger event
X5 = True - Condition variable
X7 = True - Condition variable
Expected Output: FLC

78) Test specification ALTSEL-VS-2
Prefix: X12 - Reach VS
Test case value: X3 = True - P43 trigger event
X4 - P44 trigger event
X5 = True - Condition variable
X7 = False - Condition variable

Expected Output: PITCH

79) Test specification ALTSEL-VS-3

Prefix: X12 - Reach VS
Test case value: X3 = True - P43 trigger event
X4
X6 = True - Condition variable

Expected Output: ALTHOLD

80) Test specification ALTSEL-VS-4

Prefix: X12 - Reach VS
Test case value: X3 = True - P43 trigger event
X11 - P46 trigger event

Expected Output: ALTHOLD

81) Test specification ALTSEL-VS-5

Prefix: X12 - Reach VS
Test case value: X3 = True - P43 trigger event
X12 - P52 trigger event
X7 = True - Condition variable

Expected Output: FLC

82) Test specification ALTSEL-VS-6

Prefix: X12 - Reach VS
Test case value: X3 = True - P43 trigger event
X12 - P48 trigger event
X7 = False - Condition variable

Expected Output: VS

83) Test specification ALTSEL-VS-7

Prefix: X12 - Reach VS
Test case value: X3 = True - P43 trigger event
X13 - P50 trigger event

Expected Output: FLC

84) Test specification ALTSEL-VS-8

Prefix: X12 - Reach VS
Test case value: X3 = True - P43 trigger event
X8 = APPR - P53 trigger event

Expected Output: APPR

85) Test specification ALTSEL-VS-9
<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X12</td>
<td>X3 = True</td>
<td>FLC</td>
</tr>
<tr>
<td></td>
<td>X8 = APPR</td>
<td></td>
</tr>
<tr>
<td></td>
<td>X7 = True</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

86) Test specification ALTSEL-VS-10

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X12</td>
<td>X3 = True</td>
<td>GA</td>
</tr>
<tr>
<td></td>
<td>X8 = APPR</td>
<td></td>
</tr>
<tr>
<td></td>
<td>X7 = False</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

87) Test specification ALTSEL-APPR-1

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X3 = True</td>
<td>X4</td>
<td>FLC</td>
</tr>
<tr>
<td>X8 = True</td>
<td>X5 = True</td>
<td></td>
</tr>
<tr>
<td></td>
<td>X7 = False</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

88) Test specification ALTSEL-APPR-2

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X3 = True</td>
<td>X4</td>
<td>PITCH</td>
</tr>
<tr>
<td>X8 = True</td>
<td>X5 = True</td>
<td></td>
</tr>
<tr>
<td></td>
<td>X7 = False</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

89) Test specification ALTSEL-APPR-3

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X3 = True</td>
<td>X4</td>
<td>ALTHOLD</td>
</tr>
<tr>
<td>X8 = True</td>
<td>X6 = True</td>
<td></td>
</tr>
</tbody>
</table>

90) Test specification ALTSEL-APPR-4

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X8 = True</td>
<td>X11</td>
<td>ALTHOLD</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

91) Test specification ALTSEL-APPR-5

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>Test case value:</th>
<th>Expected Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X8 = True</td>
<td>X3 = True</td>
<td>PITCH</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
X12 - P52 trigger event
X7 = True - Condition variable

Expected Output: FLC

92) Test specification ALTSEL-APPR-6
Prefix: X8 = True - Reach APPR
Test case value: X3 = True - P43 trigger event
X12 - P48 trigger event
X7 = False - Condition variable

Expected Output: VS

93) Test specification ALTSEL-APPR-7
Prefix: X8 = True - Reach APPR
Test case value: X3 = True - P43 trigger event
X13 - P50 trigger event

Expected Output: FLC

94) Test specification ALTSEL-APPR-8
Prefix: X8 = True - Reach APPR
Test case value: X3 = True - P43 trigger event
X8 = APPR - P53 trigger event

Expected Output: APPR

95) Test specification ALTSEL-APPR-9
Prefix: X8 = True - Reach APPR
Test case value: X3 = True - P43 trigger event
X14 - P52 trigger event
X7 = True - Condition variable

Expected Output: FLC

96) Test specification ALTSEL-APPR-10
Prefix: X8 = True - Reach APPR
Test case value: X3 = True - P43 trigger event
X14 - P55 trigger event
X7 = False - Condition variable

Expected Output: GA

97) Test specification ALTSEL-ALTHOLD-1
Prefix: X11 - Reach ALTHOLD
Test case value: X3 = True - P43 trigger event
X4 - P52 trigger event
X5 = True - Condition variable
X7 = True - Condition variable

Expected Output: FLC
98) Test specification ALTSEL-ALTHOLD-2

Prefix: X11 - Reach ALTHOLD
Test case value: X3 = True - P43 trigger event
X4
X5 = True - P44 trigger event
X7 = True - Condition variable
Expected Output: PITCH

99) Test specification ALTSEL-ALTHOLD-3

Prefix: X11 - Reach ALTHOLD
Test case value: X3 = True - P43 trigger event
X4 - P45 trigger event
X6 = True - Condition variable
Expected Output: ALTHOLD

100) Test specification ALTSEL-ALTHOLD-4

Prefix: X11 - Reach ALTHOLD
Test case value: X3 = True - P43 trigger event
X1 - P46 trigger event
Expected Output: ALTHOLD

101) Test specification ALTSEL-ALTHOLD-5

Prefix: X11 - Reach ALTHOLD
Test case value: X3 = True - P43 trigger event
X12 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

102) Test specification ALTSEL-ALTHOLD-6

Prefix: X11 - Reach ALTHOLD
Test case value: X3 = True - P43 trigger event
X12 - P48 trigger event
X7 = False - Condition variable
Expected Output: VS

103) Test specification ALTSEL-ALTHOLD-7

Prefix: X11 - Reach ALTHOLD
Test case value: X3 = True - P43 trigger event
X13 - P50 trigger event
Expected Output: FLC

104) Test specification ALTSEL-ALTHOLD-8

Prefix: X11 - Reach ALTHOLD
Test case value: X3 = True - P43 trigger event

X8 = APPR - P53 trigger event
Expected Output: APPR

105) Test specification ALTSEL-ALTHOLD-9

Prefix: X11 - Reach ALTHOLD
Test case value: X3 = True - P43 trigger event
X14 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

106) Test specification ALTSEL-ALTHOLD-10

Prefix: X11 - Reach ALTHOLD
Test case value: X3 = True - P43 trigger event
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

107) Test specification ALTSEL-PITCH-1

Prefix: X2 - Reach PITCH
Test case value: X3 = True - P43 trigger event
X4 - P52 trigger event
X5 = True - Condition variable
X7 = True - Condition variable
Expected Output: FLC

108) Test specification ALTSEL-PITCH-2

Prefix: X2 - Reach PITCH
Test case value: X3 = True - P43 trigger event
X4 - P54 trigger event
X5 = True - Condition variable
X7 = False - Condition variable
Expected Output: PITCH

109) Test specification ALTSEL-PITCH-3

Prefix: X2 - Reach PITCH
Test case value: X3 = True - P43 trigger event
X4 - P45 trigger event
X6 = True - Condition variable
Expected Output: ALTHOLD

110) Test specification ALTSEL-PITCH-4

Prefix: X2 - Reach PITCH
Test case value: X3 = True - P43 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD
111) Test specification ALTSEL-PITCH-5

Prefix: X2  - Reach PITCH
Test case value: X3 = True  - P43 trigger event
     X12  - P52 trigger event
     X7 = True  - Condition variable
Expected Output: FLC

112) Test specification ALTSEL-PITCH-6

Prefix: X2  - Reach PITCH
Test case value: X3 = True  - P43 trigger event
     X12  - P48 trigger event
     X7 = False  - Condition variable
Expected Output: VS

113) Test specification ALTSEL-PITCH-7

Prefix: X2  - Reach PITCH
Test case value: X3 = True  - P43 trigger event
     X13  - P50 trigger event
Expected Output: FLC

114) Test specification ALTSEL-PITCH-8

Prefix: X2  - Reach PITCH
Test case value: X3 = True  - P43 trigger event
     X8 = APPR  - P53 trigger event
Expected Output: APPR

115) Test specification ALTSEL-PITCH-9

Prefix: X2  - Reach PITCH
Test case value: X3 = True  - P43 trigger event
     X14  - P52 trigger event
     X7 = True  - Condition variable
Expected Output: FLC

116) Test specification ALTSEL-PITCH-10

Prefix: X2  - Reach PITCH
Test case value: X3 = True  - P43 trigger event
     X14  - P55 trigger event
     X7 = False  - Condition variable
Expected Output: GA

117) Test specification ALTSEL-FLC-1

Prefix: X13  - Reach FLC
Test case value: X3 = True  - P43 trigger event
     X4  - P52 trigger event
X5 = True - Condition variable
X7 = True - Condition variable

Expected Output: FLC

118) Test specification ALTSEL-FLC-2

Prefix: X13 - Reach FLC
Test case value: X3 = True - P43 trigger event
X4 - P44 trigger event
X5 = True - Condition variable
X7 = False - Condition variable

Expected Output: PITCH

119) Test specification ALTSEL-FLC-3

Prefix: X13 - Reach FLC
Test case value: X3 = True - P43 trigger event
X4 - P45 trigger event
X6 = True - Condition variable

Expected Output: ALTHOLD

120) Test specification ALTSEL-FLC-4

Prefix: X13 - Reach FLC
Test case value: X3 = True - P43 trigger event
X11 - P46 trigger event

Expected Output: ALTHOLD

121) Test specification ALTSEL-FLC-5

Prefix: X13 - Reach FLC
Test case value: X3 = True - P43 trigger event
X12 - P52 trigger event
X7 = True - Condition variable

Expected Output: FLC

122) Test specification ALTSEL-FLC-6

Prefix: X13 - Reach FLC
Test case value: X3 = True - P43 trigger event
X12 - P48 trigger event
X7 = False - Condition variable

Expected Output: VS

123) Test specification ALTSEL-FLC-7

Prefix: X13 - Reach FLC
Test case value: X3 = True - P43 trigger event
X13 - P50 trigger event

Expected Output: FLC
124) Test specification ALTSEL-FLC-8

Prefix: X13 - Reach FLC
Test case value: X3 = True - P43 trigger event
X8 = APPR - P53 trigger event
Expected Output: APPR

125) Test specification ALTSEL-FLC-9

Prefix: X13 - Reach FLC
Test case value: X3 = True - P43 trigger event
X14 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

126) Test specification ALTSEL-FLC-10

Prefix: X13 - Reach FLC
Test case value: X3 = True - P43 trigger event
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

127) Test specification ALTHOLD-ALTSEL-1

Prefix: X3 = True - Reach ALTSEL
Test case value: X4 = True - P45 trigger event
X6 = True - Condition variable
X2 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

128) Test specification ALTHOLD-ALTSEL-2

Prefix: X3 = True - Reach ALTSEL
Test case value: X4 = True - P45 trigger event
X6 = True - Condition variable
X2 - P42 trigger event
X7 = False - Condition variable
Expected Output: PITCH

129) Test specification ALTHOLD-ALTSEL-3

Prefix: X3 = True - Reach ALTSEL
Test case value: X4 = True - P45 trigger event
X6 = True - Condition variable
X3 = True - P43 trigger event
Expected Output: ALTSEL

130) Test specification ALTHOLD-ALTSEL-4
<table>
<thead>
<tr>
<th>Test Case</th>
<th>Description</th>
</tr>
</thead>
</table>
| 131) Test specification ALTHOLD-ALTSEL-5 | Prefix: X3 = True - Reach ALTSEL  
Test case value: X4 - P45 trigger event  
X6 = True - Condition variable  
X12 - P52 trigger event  
X7 = True - Condition variable  
Expected Output: FLC |
| 132) Test specification ALTHOLD-ALTSEL-6 | Prefix: X3 = True - Reach ALTSEL  
Test case value: X4 - P45 trigger event  
X6 = True - Condition variable  
X13 - P52 trigger event  
Expected Output: PITCH |
| 133) Test specification ALTHOLD-ALTSEL-7 | Prefix: X3 = True - Reach ALTSEL  
Test case value: X4 - P45 trigger event  
X6 = True - Condition variable  
X8 = True - P53 trigger event  
Expected Output: APPR |
| 134) Test specification ALTHOLD-ALTSEL-8 | Prefix: X3 = True - Reach ALTSEL  
Test case value: X4 - P45 trigger event  
X6 = True - Condition variable  
X11 - P46 trigger event  
X7 = True - Condition variable  
Expected Output: FLC |
| 135) Test specification ALTHOLD-ALTSEL-9 | Prefix: X3 = True - Reach ALTSEL  
Test case value: X4 - P45 trigger event  
X6 = True - Condition variable  
X11 - P47 trigger event  
X7 = False - Condition variable  
Expected Output: PITCH |
| 136) Test specification ALTHOLD-ALTSEL-10 |
Prefix: \( X_3 = \text{True} \) - Reach ALTSEL
Test case value: \( X_4 \) - P45 trigger event
\( X_6 = \text{True} \) - Condition variable
\( X_{14} \) - P55 trigger event
\( X_7 = \text{True} \) - Condition variable
Expected Output: FLC

137) Test specification ALTHOLD-ALTSEL-11

Prefix: \( X_3 = \text{True} \) - Reach ALTSEL
Test case value: \( X_4 \) - P45 trigger event
\( X_6 = \text{True} \) - Condition variable
\( X_{14} \) - P55 trigger event
\( X_7 = \text{False} \) - Condition variable
Expected Output: GA

138) Test specification ALTHOLD-GA-1

Prefix: \( X_{14} \) - Reach GA
Test case value: \( X_{11} \) - P46 trigger event
\( X_2 \) - P42 trigger event
\( X_7 = \text{True} \) - Condition variable
Expected Output: FLC

139) Test specification ALTHOLD-GA-2

Prefix: \( X_{14} \) - Reach GA
Test case value: \( X_{11} \) - P46 trigger event
\( X_2 \) - P42 trigger event
\( X_7 = \text{False} \) - Condition variable
Expected Output: PITCH

140) Test specification ALTHOLD-GA-3

Prefix: \( X_{14} \) - Reach GA
Test case value: \( X_{11} \) - P46 trigger event
\( X_3 = \text{True} \) - P43 trigger event
Expected Output: ALTSEL

141) Test specification ALTHOLD-GA-4

Prefix: \( X_{14} \) - Reach GA
Test case value: \( X_{11} \) - P46 trigger event
\( X_{12} \) - P52 trigger event
\( X_7 = \text{True} \) - Condition variable
Expected Output: FLC

142) Test specification ALTHOLD-GA-5

Prefix: \( X_{14} \) - Reach GA
Test case value: \( X_{11} \) - P46 trigger event
X12 - P50 trigger event
X7 = False - Condition variable

Expected Output: PITCH

143) Test specification ALTHOLD-GA-6
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X13 - P52 trigger event

Expected Output: FLC

144) Test specification ALTHOLD-GA-7
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X8 = True - P53 trigger event

Expected Output: APR

145) Test specification ALTHOLD-GA-8
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X11 - P46 trigger event
X7 = True - Condition variable

Expected Output: FLC

146) Test specification ALTHOLD-GA-9
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X14 - P47 trigger event
X7 = False - Condition variable

Expected Output: PITCH

147) Test specification ALTHOLD-GA-10
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X14 - P55 trigger event
X7 = True - Condition variable

Expected Output: FLC

148) Test specification ALTHOLD-GA-11
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X14 - P55 trigger event
X7 = False - Condition variable

Expected Output: GA

149) Test specification ALTHOLD-VS-1
Prefix: X12 - Reach VS
Test case value: X11 - P46 trigger event
X2 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

150) Test specification ALTHOLD-VS-2
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X2 - P42 trigger event
X7 = False - Condition variable
Expected Output: PITCH

151) Test specification ALTHOLD-VS-3
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

152) Test specification ALTHOLD-VS-4
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X12 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

153) Test specification ALTHOLD-VS-5
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X12 - P50 trigger event
X7 = False - Condition variable
Expected Output: PITCH

154) Test specification ALTHOLD-VS-6
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X13 - P52 trigger event
Expected Output: FLC

155) Test specification ALTHOLD-VS-7
Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
X8 = True - P53 trigger event
Expected Output: APPR
156) Test specification ALTHOLD-VS-8

Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
 X11 - P46 trigger event
 X7 = True - Condition variable
Expected Output: FLC

157) Test specification ALTHOLD-VS-9

Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
 X11 - P47 trigger event
 X7 = False - Condition variable
Expected Output: PITCH

158) Test specification ALTHOLD-VS-10

Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
 X14 - P55 trigger event
 X7 = True - Condition variable
Expected Output: FLC

159) Test specification ALTHOLD-VS-11

Prefix: X14 - Reach GA
Test case value: X11 - P46 trigger event
 X14 - P55 trigger event
 X7 = False - Condition variable
Expected Output: GA

160) Test specification ALTHOLD-PITCH-1

Prefix: X2 - Reach PITCH
Test case value: X11 - P46 trigger event
 X2 = True - P52 trigger event
 X7 = True - Condition variable
Expected Output: FLC

161) Test specification ALTHOLD-PITCH-2

Prefix: X2 - Reach PITCH
Test case value: X11 - P46 trigger event
 X2 - P42 trigger event
 X7 = False - Condition variable
Expected Output: PITCH

162) Test specification ALTHOLD-PITCH-3

Prefix: X2 - Reach PITCH
Test case value: X11 - P46 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

163) Test specification ALTHOLD-PITCH-4
Prefix: X2 - Reach PITCH
Test case value: X11 - P46 trigger event
X12 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

164) Test specification ALTHOLD-PITCH-5
Prefix: X2 - Reach PITCH
Test case value: X11 - P46 trigger event
X12 - P50 trigger event
X7 = False - Condition variable
Expected Output: PITCH

165) Test specification ALTHOLD-PITCH-6
Prefix: X2 - Reach PITCH
Test case value: X11 - P46 trigger event
X13 - P52 trigger event
Expected Output: FLC

166) Test specification ALTHOLD-PITCH-7
Prefix: X2 - Reach PITCH
Test case value: X11 - P46 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

167) Test specification ALTHOLD-PITCH-8
Prefix: X2 - Reach PITCH
Test case value: X11 - P46 trigger event
X11 - P46 trigger event
X7 = True - Condition variable
Expected Output: FLC

168) Test specification ALTHOLD-PITCH-9
Prefix: X2 - Reach PITCH
Test case value: X11 - P46 trigger event
X11 - P47 trigger event
X7 = False - Condition variable
Expected Output: PITCH

169) Test specification ALTHOLD-PITCH-10
170) Test specification ALTHOLD-PITCH-11

Prefix: X2 - Reach PITCH
Test case value: X11 - P46 trigger event
X14 - P55 trigger event
X7 = True - Condition variable
Expected Output: FLC

171) Test specification ALTHOLD-ALTSEL-1

Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - P46 trigger event
X2 - P52 trigger event
X7 = True - Condition variable
Expected Output: GA

172) Test specification ALTHOLD-ALTSEL-2

Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - P46 trigger event
X2 - P42 trigger event
X7 = False - Condition variable
Expected Output: PITCH

173) Test specification ALTHOLD-ALTSEL-3

Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - P46 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

174) Test specification ALTHOLD-ALTSEL-4

Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - P46 trigger event
X12 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

175) Test specification ALTHOLD-ALTSEL-5

Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - P46 trigger event
X12 - P50 trigger event
X7 = False - Condition variable
Expected Output: PITCH

176) Test specification ALTHOLD-ALTSEL-6
Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - P46 trigger event
X13 - P52 trigger event
Expected Output: FLC

177) Test specification ALTHOLD-ALTSEL-7
Prefix: X3 = True - Reach ALTSEL
Test case value: X8 = True - P46 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

178) Test specification ALTHOLD-ALTSEL-8
Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - P46 trigger event
X11 - P46 trigger event
X7 = True - Condition variable
Expected Output: FLC

179) Test specification ALTHOLD-ALTSEL-9
Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - P46 trigger event
X11 - P47 trigger event
X7 = False - Condition variable
Expected Output: PITCH

180) Test specification ALTHOLD-ALTSEL-10
Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - P46 trigger event
X14 - P55 trigger event
X7 = True - Condition variable
Expected Output: FLC

181) Test specification ALTHOLD-ALTSEL-11
Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - P46 trigger event
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

182) Test specification ALTHOLD-FLC-1
Prefix: X13 - Reach FLC
Test case value: X11 - P46 trigger event
X2 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

183) Test specification ALTHOLD-FLC-2

Prefix: X13 - Reach FLC
Test case value: X11 - P46 trigger event
X2 - P42 trigger event
X7 = False - Condition variable
Expected Output: PITCH

184) Test specification ALTHOLD-FLC-3

Prefix: X13 - Reach FLC
Test case value: X11 - P46 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

185) Test specification ALTHOLD-FLC-4

Prefix: X13 - Reach FLC
Test case value: X11 - P46 trigger event
X12 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

186) Test specification ALTHOLD-FLC-5

Prefix: X13 - Reach FLC
Test case value: X11 - P46 trigger event
X12 - P50 trigger event
X7 = False - Condition variable
Expected Output: PITCH

187) Test specification ALTHOLD-FLC-6

Prefix: X13 - Reach FLC
Test case value: X11 - P46 trigger event
X13 - P52 trigger event
Expected Output: FLC

188) Test specification ALTHOLD-FLC-7

Prefix: X13 - Reach FLC
Test case value: X11 - P46 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

189) Test specification ALTHOLD-FLC-8
Prefix: X13 - Reach FLC
Test case value: X11 - P46 trigger event
X11 - P46 trigger event
X7 = True - Condition variable
Expected Output: FLC

190) Test specification ALTHOLD-FLC-9
Prefix: X13 - Reach FLC
Test case value: X11 - P46 trigger event
X11 - P47 trigger event
X7 = False - Condition variable
Expected Output: PITCH

191) Test specification ALTHOLD-FLC-10
Prefix: X13 - Reach FLC
Test case value: X11 - P46 trigger event
X14 - P55 trigger event
X7 = True - Condition variable
Expected Output: FLC

192) Test specification ALTHOLD-FLC-11
Prefix: X13 - Reach FLC
Test case value: X11 - P46 trigger event
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

193) Test specification VS-GA-1
Prefix: X14 - Reach GA
Test case value: X12 - P48 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

194) Test specification VS-GA-2
Prefix: X14 - Reach GA
Test case value: X12 - P48 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

195) Test specification VS-GA-3
Prefix: X14 - Reach GA
Test case value: X12 - P48 trigger event
X12 - P49 trigger event
Expected Output: PITCH
196) Test specification VS-GA-4

Prefix: X14 - Reach GA
Test case value: X12 - P48 trigger event
X13 - P50 trigger event
Expected Output: FLC

197) Test specification VS-GA-5

Prefix: X14 - Reach GA
Test case value: X12 - P48 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

198) Test specification VS-GA-6

Prefix: X14 - Reach GA
Test case value: X12 - P48 trigger event
X14 - P55 trigger event
Expected Output: GA

199) Test specification VS-ALTSEL-1

Prefix: X3 = True - Reach ALTSEL
Test case value: X12 - P48 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

200) Test specification VS-ALTSEL-2

Prefix: X3 = True - Reach ALTSEL
Test case value: X12 - P48 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

201) Test specification VS-ALTSEL-3

Prefix: X3 = True - Reach ALTSEL
Test case value: X12 - P48 trigger event
X12 - P49 trigger event
Expected Output: PITCH

202) Test specification VS-ALTSEL-4

Prefix: X3 = True - Reach ALTSEL
Test case value: X12 - P48 trigger event
X13 - P50 trigger event
Expected Output: FLC

203) Test specification VS-ALTSEL-5
Prefix: X3 = True - Reach ALTSEL
Test case value: X12 - P48 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

204) Test specification VS-ALTSEL-6
Prefix: X3 = True - Reach ALTSEL
Test case value: X12 - P48 trigger event
X14 - P55 trigger event
Expected Output: GA

205) Test specification VS-ALTHOLD-1
Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P48 trigger event
X3 = True - P43 trigger event
Expected Output: ALTHOLD

206) Test specification VS-ALTHOLD-2
Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P48 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

207) Test specification VS-ALTHOLD-3
Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P48 trigger event
X12 - P49 trigger event
Expected Output: PITCH

208) Test specification VS-ALTHOLD-4
Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P48 trigger event
X13 - P50 trigger event
Expected Output: FLC

209) Test specification VS-ALTHOLD-5
Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P48 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

210) Test specification VS-ALTHOLD-6
Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P48 trigger event
X14 - P55 trigger event

Expected Output: GA

211) Test specification VS-PITCH-1

Prefix: X2 - Reach PITCH
Test case value: X12 - P48 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

212) Test specification VS-PITCH-2

Prefix: X2 - Reach PITCH
Test case value: X12 - P48 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

213) Test specification VS-PITCH-3

Prefix: X2 - Reach PITCH
Test case value: X12 - P48 trigger event
X12 - P49 trigger event
Expected Output: PITCH

214) Test specification VS-PITCH-4

Prefix: X2 - Reach PITCH
Test case value: X12 - P48 trigger event
X13 - P50 trigger event
Expected Output: FLC

215) Test specification VS-PITCH-5

Prefix: X2 - Reach PITCH
Test case value: X12 - P48 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

216) Test specification VS-PITCH-6

Prefix: X2 - Reach PITCH
Test case value: X12 - P48 trigger event
X14 - P55 trigger event
Expected Output: GA

217) Test specification VS-FLC-1

Prefix: X13 - Reach FLC
Test case value: X12 - P48 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL
218) Test specification VS-FLC-2

Prefix: X13 - Reach FLC
Test case value: X12 - P48 trigger event
X11 - P46 trigger event
Expected Output: ALTHROLD

219) Test specification VS-FLC-3

Prefix: X13 - Reach FLC
Test case value: X12 - P48 trigger event
X12 - P49 trigger event
Expected Output: PITCH

220) Test specification VS-FLC-4

Prefix: X13 - Reach FLC
Test case value: X12 - P48 trigger event
X13 - P50 trigger event
Expected Output: FLC

221) Test specification VS-FLC-5

Prefix: X13 - Reach FLC
Test case value: X12 - P48 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

222) Test specification VS-FLC-6

Prefix: X13 - Reach FLC
Test case value: X12 - P48 trigger event
X14 - P55 trigger event
Expected Output: GA

223) Test specification FLC-GA-1

Prefix: X14 - Reach GA
Test case value: X13 - P50 trigger event
X2 - P42 trigger event
X7 = False - Condition variable
Expected Output: GA

224) Test specification FLC-GA-2

Prefix: X14 - Reach GA
Test case value: X13 - P50 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

225) Test specification FLC-GA-3
Prefix: X14 - Reach GA
Test case value: X13 - P50 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

226) Test specification FLC-GA-4

Prefix: X14 - Reach GA
Test case value: X13 - P50 trigger event
X12 - P48 trigger event
X7 = False - Condition variable
Expected Output: PITCH

227) Test specification FLC-GA-5

Prefix: X14 - Reach GA
Test case value: X13 - P50 trigger event
X13 - P49 trigger event
X7 = False - Condition variable
Expected Output: PITCH

228) Test specification FLC-GA-6

Prefix: X14 - Reach GA
Test case value: X13 - P50 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

229) Test specification FLC-GA-7

Prefix: X14 - Reach GA
Test case value: X13 - P50 trigger event
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

230) Test specification FLC-VS-1

Prefix: X12 - Reach VS
Test case value: X13 - P50 trigger event
X2 - P42 trigger event
X7 = False - Condition variable
Expected Output: GA

231) Test specification FLC-VS-2

Prefix: X12 - Reach VS
Test case value: X13 - P50 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL
232) Test specification FLC-VS-3

Prefix: X12 - Reach VS
Test case value: X13 - P50 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

233) Test specification FLC-VS-4

Prefix: X12 - Reach VS
Test case value: X13 - P50 trigger event
X12 - P48 trigger event
X7 = False - Condition variable
Expected Output: VS

234) Test specification FLC-VS-5

Prefix: X12 - Reach VS
Test case value: X13 - P50 trigger event
X13 - P49 trigger event
X7 = False - Condition variable
Expected Output: PITCH

235) Test specification FLC-VS-6

Prefix: X12 - Reach VS
Test case value: X13 - P50 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

236) Test specification FLC-VS-7

Prefix: X12 - Reach VS
Test case value: X13 - P50 trigger event
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

237) Test specification FLC-ALTSEL-1

Prefix: X3 = True - Reach ALTSEL
Test case value: X13 - P50 trigger event
X2 - P42 trigger event
X7 = False - Condition variable
Expected Output: GA

238) Test specification FLC-ALTSEL-2

Prefix: X3 = True - Reach ALTSEL
Test case value: X13 - P50 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

239) Test specification FLC-ALTSEL-3

Prefix: X3 = True
Test case value: X13
       X11
Expected Output: ALTHOLD

240) Test specification FLC-ALTSEL-4

Prefix: X3 = True
Test case value: X13
       X12
       X7 = False
Expected Output: VS

241) Test specification FLC-ALTSEL-5

Prefix: X3 = True
Test case value: X13
       X13
       X7 = False
Expected Output: PITCH

242) Test specification FLC-ALTSEL-6

Prefix: X3 = True
Test case value: X13
       X8 = True
Expected Output: APPR

243) Test specification FLC-ALTSEL-7

Prefix: X3 = True
Test case value: X13
       X14
       X7 = False
Expected Output: GA

244) Test specification FLC-ALTSEL-8

Prefix: X3 = True
Test case value: X4
       X7 = True
       X2
       X7 = False
Expected Output: GA

245) Test specification FLC-ALTSEL-9
Prefix: X3 = True - Reach ALTSEL
Test case value: X4 - P52 trigger event
X7 = True - Condition variable
X3 = True - P43 trigger event
Expected Output: ALTSEL

246) Test specification FLC-ALTSEL-10

Prefix: X3 = True - Reach ALTSEL
Test case value: X4 - P52 trigger event
X7 = True - Condition variable
X11 - P46 trigger event
Expected Output: ALTHOLD

247) Test specification FLC-ALTSEL-11

Prefix: X3 = True - Reach ALTSEL
Test case value: X4 - P52 trigger event
X7 = True - Condition variable
X12 - P48 trigger event
X7 = False - Condition variable
Expected Output: VS

248) Test specification FLC-ALTSEL-12

Prefix: X3 = True - Reach ALTSEL
Test case value: X4 - P52 trigger event
X7 = True - Condition variable
X13 - P49 trigger event
X7 = False - Condition variable
Expected Output: PITCH

249) Test specification FLC-ALTSEL-13

Prefix: X3 = True - Reach ALTSEL
Test case value: X4 - P52 trigger event
X7 = True - Condition variable
X8 = True - P53 trigger event
Expected Output: APPR

250) Test specification FLC-ALTSEL-14

Prefix: X3 = True - Reach ALTSEL
Test case value: X4 - P52 trigger event
X7 = True - Condition variable
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

251) Test specification FLC-ALTSEL-15
<table>
<thead>
<tr>
<th>Prefix: X3 = True</th>
<th>- Reach ALTSEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test case value: X11</td>
<td>- P52 trigger event</td>
</tr>
<tr>
<td>X7 = True</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>X2</td>
<td>- P42 trigger event</td>
</tr>
<tr>
<td>X7 = False</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>Expected Output: GA</td>
<td></td>
</tr>
</tbody>
</table>

252) Test specification FLC-ALTSEL-16

<table>
<thead>
<tr>
<th>Prefix: X3 = True</th>
<th>- Reach ALTSEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test case value: X11</td>
<td>- P52 trigger event</td>
</tr>
<tr>
<td>X7 = True</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>X3 = True</td>
<td>- P43 trigger event</td>
</tr>
<tr>
<td>Expected Output: ALTSEL</td>
<td></td>
</tr>
</tbody>
</table>

253) Test specification FLC-ALTSEL-17

<table>
<thead>
<tr>
<th>Prefix: X3 = True</th>
<th>- Reach ALTSEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test case value: X11</td>
<td>- P52 trigger event</td>
</tr>
<tr>
<td>X7 = True</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>X11</td>
<td>- P46 trigger event</td>
</tr>
<tr>
<td>Expected Output: ALTHOLD</td>
<td></td>
</tr>
</tbody>
</table>

254) Test specification FLC-ALTSEL-18

<table>
<thead>
<tr>
<th>Prefix: X3 = True</th>
<th>- Reach ALTSEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test case value: X11</td>
<td>- P52 trigger event</td>
</tr>
<tr>
<td>X7 = True</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>X12</td>
<td>- P48 trigger event</td>
</tr>
<tr>
<td>X7 = False</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>Expected Output: VS</td>
<td></td>
</tr>
</tbody>
</table>

255) Test specification FLC-ALTSEL-19

<table>
<thead>
<tr>
<th>Prefix: X3 = True</th>
<th>- Reach ALTSEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test case value: X11</td>
<td>- P52 trigger event</td>
</tr>
<tr>
<td>X7 = True</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>X13</td>
<td>- P49 trigger event</td>
</tr>
<tr>
<td>X7 = False</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>Expected Output: PITCH</td>
<td></td>
</tr>
</tbody>
</table>

256) Test specification FLC-ALTSEL-20

<table>
<thead>
<tr>
<th>Prefix: X3 = True</th>
<th>- Reach ALTSEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test case value: X11</td>
<td>- P52 trigger event</td>
</tr>
<tr>
<td>X7 = True</td>
<td>- Condition variable</td>
</tr>
<tr>
<td>X8 = True</td>
<td>- P53 trigger event</td>
</tr>
<tr>
<td>Expected Output: APPR</td>
<td></td>
</tr>
</tbody>
</table>

257) Test specification FLC-ALTSEL-21
Prefix: X3 = True - Reach ALTSEL
Test case value: X11 - P52 trigger event
X7 = True - Condition variable
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

258) Test specification FLC-ALTSEL-22

Prefix: X3 = True - Reach ALTSEL
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X2 - P42 trigger event
X7 = False - Condition variable
Expected Output: GA

259) Test specification FLC-ALTSEL-23

Prefix: X3 = True - Reach ALTSEL
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X3 = True - P43 trigger event
Expected Output: ALTSEL

260) Test specification FLC-ALTSEL-24

Prefix: X3 = True - Reach ALTSEL
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X11 - P46 trigger event
Expected Output: ALTHOLD

261) Test specification FLC-ALTSEL-25

Prefix: X3 = True - Reach ALTSEL
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X12 - P48 trigger event
X7 = False - Condition variable
Expected Output: VS

262) Test specification FLC-ALTSEL-26

Prefix: X3 = True - Reach ALTSEL
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X13 - P49 trigger event
X7 = False - Condition variable
Expected Output: PITCH

263) Test specification FLC-ALTSEL-27
Prefix: X3 = True - Reach ALTSEL
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X8 = True - P53 trigger event
Expected Output: APPR

264) Test specification FLC-ALTSEL-28
Prefix: X3 = True - Reach ALTSEL
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X14 - P55 trigger event
X7 = False - Condition variable

Expected Output: GA

265) Test specification FLC-ALTHOLD-1
Prefix: X11 - Reach ALTHOLD
Test case value: X13 - P50 trigger event
X2 - P42 trigger event
X7 = False - Condition variable

Expected Output: GA

266) Test specification FLC-ALTHOLD-2
Prefix: X11 - Reach ALTHOLD
Test case value: X13 - P50 trigger event
X3 = True - P43 trigger event

Expected Output: ALTSEL

267) Test specification FLC-ALTHOLD-3
Prefix: X11 - Reach ALTHOLD
Test case value: X13 - P50 trigger event
X11 - P46 trigger event

Expected Output: ALTHOLD

268) Test specification FLC-ALTHOLD-4
Prefix: X11 - Reach ALTHOLD
Test case value: X13 - P50 trigger event
X12 - P48 trigger event
X7 = False - Condition variable

Expected Output: VS

269) Test specification FLC-ALTHOLD-5
Prefix: X11 - Reach ALTHOLD
Test case value: X13 - P50 trigger event
X13 - P49 trigger event
X7 = False - Condition variable
Expected Output: PITCH

270) Test specification FLC-ALTHOLD-6

Prefix: X11 - Reach ALTHOLD
Test case value: X13 - P50 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

271) Test specification FLC-ALTHOLD-7

Prefix: X11 - Reach ALTHOLD
Test case value: X13 - P52 trigger event
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

272) Test specification FLC-ALTHOLD-8

Prefix: X11 - Reach ALTHOLD
Test case value: X2 - P52 trigger event
X7 = True - Condition variable
X2 - P42 trigger event
X7 = False - Condition variable
Expected Output: GA

273) Test specification FLC-ALTHOLD-9

Prefix: X11 - Reach ALTHOLD
Test case value: X2 - P52 trigger event
X7 = True - Condition variable
X3 = True - P43 trigger event
Expected Output: ALTSEL

274) Test specification FLC-ALTHOLD-10

Prefix: X11 - Reach ALTHOLD
Test case value: X2 - P52 trigger event
X7 = True - Condition variable
X11 - P46 trigger event
Expected Output: ALTHOLD

275) Test specification FLC-ALTHOLD-11

Prefix: X11 - Reach ALTHOLD
Test case value: X2 - P52 trigger event
X7 = True - Condition variable
X12 - P48 trigger event
X7 = False - Condition variable
Expected Output: VS

164
276) Test specification FLC-ALTHOLD-12

Prefix: X11  - Reach ALTHOLD
Test case value: X2  - P52 trigger event
    X7 = True  - Condition variable
    X13  - P49 trigger event
    X7 = False  - Condition variable
Expected Output: PITCH

277) Test specification FLC-ALTHOLD-13

Prefix: X11  - Reach ALTHOLD
Test case value: X2  - P52 trigger event
    X7 = True  - Condition variable
    X8 = True  - P53 trigger event
Expected Output: APPR

278) Test specification FLC-ALTHOLD-14

Prefix: X11  - Reach ALTHOLD
Test case value: X2  - P52 trigger event
    X7 = True  - Condition variable
    X14  - P55 trigger event
    X7 = False  - Condition variable
Expected Output: GA

279) Test specification FLC-ALTHOLD-15

Prefix: X11  - Reach ALTHOLD
Test case value: X11  - P52 trigger event
    X7 = True  - Condition variable
    X2  - P42 trigger event
    X7 = False  - Condition variable
Expected Output: GA

280) Test specification FLC-ALTHOLD-16

Prefix: X11  - Reach ALTHOLD
Test case value: X11  - P52 trigger event
    X7 = True  - Condition variable
    X3 = True  - P43 trigger event
Expected Output: ALTSEL

281) Test specification FLC-ALTHOLD-17

Prefix: X11  - Reach ALTHOLD
Test case value: X11  - P52 trigger event
    X7 = True  - Condition variable
    X11  - P46 trigger event
Expected Output: ALTHOLD
282) Test specification FLC-ALTHOLD-18

Prefix: X11 - Reach ALTHOLD
Test case value: X11 - P52 trigger event
X7 = True - Condition variable
X12 - P48 trigger event
X7 = False - Condition variable
Expected Output: VS

283) Test specification FLC-ALTHOLD-19

Prefix: X11 - Reach ALTHOLD
Test case value: X11 - P52 trigger event
X7 = True - Condition variable
X13 - P49 trigger event
X7 = False - Condition variable
Expected Output: PITCH

284) Test specification FLC-ALTHOLD-20

Prefix: X11 - Reach ALTHOLD
Test case value: X11 - P52 trigger event
X7 = True - Condition variable
X14 - P53 trigger event
Expected Output: APPR

285) Test specification FLC-ALTHOLD-21

Prefix: X11 - Reach ALTHOLD
Test case value: X11 - P52 trigger event
X7 = True - Condition variable
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

286) Test specification FLC-ALTHOLD-22

Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P52 trigger event
X7 = True - Condition variable
X2 - P42 trigger event
X7 = False - Condition variable
Expected Output: GA

287) Test specification FLC-ALTHOLD-23

Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P52 trigger event
X7 = True - Condition variable
X3 = True - P43 trigger event
Expected Output: ALTSEL
288) Test specification FLC-ALTHOLD-24

Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P52 trigger event
X7 = True - Condition variable
X11 - P46 trigger event
Expected Output: ALTHOLD

289) Test specification FLC-ALTHOLD-25

Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P52 trigger event
X7 = True - Condition variable
X12 - P48 trigger event
X7 = False - Condition variable
Expected Output: VS

290) Test specification FLC-ALTHOLD-26

Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P52 trigger event
X7 = True - Condition variable
X13 - P49 trigger event
X7 = False - Condition variable
Expected Output: PITCH

291) Test specification FLC-ALTHOLD-27

Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P52 trigger event
X7 = True - Condition variable
X8 = True - P53 trigger event
Expected Output: APPR

292) Test specification FLC-ALTHOLD-28

Prefix: X11 - Reach ALTHOLD
Test case value: X12 - P52 trigger event
X7 = True - Condition variable
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

293) Test specification FLC-ALTHOLD-29

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X2 - P42 trigger event
X7 = False - Condition variable
Expected Output: GA
294) Test specification FLC-ALTHOLD-30

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X3 = True - P43 trigger event
Expected Output: ALTSEL

295) Test specification FLC-ALTHOLD-31

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X11 - P46 trigger event
Expected Output: ALTHOLD

296) Test specification FLC-ALTHOLD-32

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X12 - P48 trigger event
X7 = False - Condition variable
Expected Output: VS

297) Test specification FLC-ALTHOLD-33

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X13 - P49 trigger event
X7 = False - Condition variable
Expected Output: PITCH

298) Test specification FLC-ALTHOLD-34

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X8 = True - P53 trigger event
Expected Output: APPR

299) Test specification FLC-ALTHOLD-35

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P52 trigger event
X7 = True - Condition variable
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA
300) Test specification FLC-PITCH-1

Prefix: X2 - Reach PITCH
Test case value: X13 - P50 trigger event
X2 = True - P42 trigger event
X1 = False - Condition variable
Expected Output: GA

301) Test specification FLC-PITCH-2

Prefix: X2 - Reach PITCH
Test case value: X13 - P50 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

302) Test specification FLC-PITCH-3

Prefix: X2 - Reach PITCH
Test case value: X13 - P50 trigger event
X1 = True - P46 trigger event
Expected Output: ALTHOLD

303) Test specification FLC-PITCH-4

Prefix: X2 - Reach PITCH
Test case value: X13 - P50 trigger event
X2 = True - P48 trigger event
X7 = False - Condition variable
Expected Output: VS

304) Test specification FLC-PITCH-5

Prefix: X2 - Reach PITCH
Test case value: X13 - P50 trigger event
X2 = True - P49 trigger event
X7 = False - Condition variable
Expected Output: PITCH

305) Test specification FLC-PITCH-6

Prefix: X2 - Reach PITCH
Test case value: X13 - P52 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

306) Test specification FLC-PITCH-7

Prefix: X2 - Reach PITCH
Test case value: X13 - P52 trigger event
X14 - P55 trigger event
X7 = False - Condition variable
Expected Output: GA

307) Test specification FLC-APPR-1

Prefix: X8 = True - Reach APPR
Test case value: X8 = True - P52 trigger event
               NOT X14 - P52 trigger event
               X7 = True - Condition variable
               X2               - Condition variable
               X7 = False - Condition variable

Expected Output: GA

308) Test specification FLC-APPR-2

Prefix: X2 - Reach PITCH
Test case value: X8 = True - P52 trigger event
               NOT X14 - P52 trigger event
               X7 = True - Condition variable
               X3 = True - P43 trigger event

Expected Output: ALTSEL

309) Test specification FLC-APPR-3

Prefix: X2 - Reach PITCH
Test case value: X8 = True - P52 trigger event
               NOT X14 - P52 trigger event
               X7 = True - Condition variable
               X11 - P46 trigger event

Expected Output: ALTHOLD

310) Test specification FLC-APPR-4

Prefix: X2 - Reach PITCH
Test case value: X8 = True - P52 trigger event
               NOT X14 - P52 trigger event
               X7 = True - Condition variable
               X12 - P48 trigger event
               X7 = False - Condition variable

Expected Output: VS

311) Test specification FLC-APPR-5

Prefix: X2 - Reach PITCH
Test case value: X8 = True - P52 trigger event
               NOT X14 - P52 trigger event
               X7 = True - Condition variable
               X13 - P49 trigger event
               X7 = False - Condition variable

Expected Output: PITCH

312) Test specification FLC-APPR-6
Prefix: X2 - Reach PITCH
Test case value: X8 = True - P52 trigger event
       NOT X14 - P52 trigger event
       X7 = True - Condition variable
       X8 = True - P53 trigger event
Expected Output: APPR

313) Test specification FLC-APPR-7

Prefix: X2 - Reach PITCH
Test case value: X8 = True - P52 trigger event
       NOT X14 - P52 trigger event
       X7 = True - Condition variable
       X14 - P55 trigger event
       X7 = False - Condition variable

314) Test specification FLC-APPR-8

Prefix: X8 = True - Reach APPR
Test case value: X14 - P52 trigger event
       X7 = True - Condition variable
       X2 - P42 trigger event
       X7 = False - Condition variable

315) Test specification FLC-APPR-9

Prefix: X2 - Reach PITCH
Test case value: X14 - P52 trigger event
       X7 = True - Condition variable
       X3 = True - P43 trigger event

316) Test specification FLC-APPR-10

Prefix: X2 - Reach PITCH
Test case value: X14 - P52 trigger event
       X7 = True - Condition variable
       X11 - P46 trigger event

317) Test specification FLC-APPR-11

Prefix: X2 - Reach PITCH
Test case value: X14 - P52 trigger event
       X7 = True - Condition variable
       X12 - P48 trigger event
       X7 = False - Condition variable

318) Test specification FLC-APPR-12
<table>
<thead>
<tr>
<th>Prefix</th>
<th>Test case value</th>
<th>Expected Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>X2</td>
<td>X14</td>
<td>PITCH</td>
</tr>
<tr>
<td>X7 = True</td>
<td>X13</td>
<td></td>
</tr>
<tr>
<td>X7 = False</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

319) Test specification FLC-APPR-13

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Test case value</th>
<th>Expected Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>X2</td>
<td>X14</td>
<td>APPR</td>
</tr>
<tr>
<td>X7 = True</td>
<td>X13</td>
<td></td>
</tr>
<tr>
<td>X7 = False</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

320) Test specification FLC-APPR-14

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Test case value</th>
<th>Expected Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>X2</td>
<td>X14</td>
<td>APPR</td>
</tr>
<tr>
<td>X7 = True</td>
<td>X13</td>
<td></td>
</tr>
<tr>
<td>X7 = False</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

321) Test specification APPR-GA-1

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Test case value</th>
<th>Expected Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>X14</td>
<td>X3</td>
<td>ALTSEL</td>
</tr>
<tr>
<td>X8 = True</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

322) Test specification APPR-GA-2

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Test case value</th>
<th>Expected Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>X14</td>
<td>X14</td>
<td>FLC</td>
</tr>
<tr>
<td>X8 = True</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOT X14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X7 = True</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

323) Test specification APPR-GA-3

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Test case value</th>
<th>Expected Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>X14 = True</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X8 = True</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOT X14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X7 = False</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

324) Test specification APPR-GA-4
Prefix: \( X_{14} = \text{True} \) - Reach \( \text{GA} \)
Test case value: \( X_8 = \text{True} \) - \( P_{53} \) trigger event
\( X_{14} \) - \( P_{52} \) trigger event
\( X_7 = \text{True} \) - Condition variable
Expected Output: \( \text{FLC} \)

325) Test specification APPR-GA-5

Prefix: \( X_{14} \) - Reach \( \text{GA} \)
Test case value: \( X_8 = \text{True} \) - \( P_{53} \) trigger event
\( X_{14} \) - \( P_{55} \) trigger event
\( X_7 = \text{True} \) - Condition variable
Expected Output: \( \text{GA} \)

326) Test specification APPR-VS-1

Prefix: \( X_{12} \) - Reach \( \text{VS} \)
Test case value: \( X_8 = \text{True} \) - \( P_{53} \) trigger event
\( X_3 = \text{True} \) - \( P_{43} \) trigger event
Expected Output: \( \text{ALTSEL} \)

327) Test specification APPR-VS-2

Prefix: \( X_{12} \) - Reach \( \text{VS} \)
Test case value: \( X_8 = \text{True} \) - \( P_{53} \) trigger event
\( X_8 = \text{True} \) - \( P_{52} \) trigger event
\( \text{NOT} \ X_{14} \) - \( P_{52} \) trigger event
\( X_7 = \text{True} \) - Condition variable
Expected Output: \( \text{FLC} \)

328) Test specification APPR-VS-3

Prefix: \( X_{12} \) - Reach \( \text{VS} \)
Test case value: \( X_8 = \text{True} \) - \( P_{53} \) trigger event
\( X_8 = \text{True} \) - \( P_{52} \) trigger event
\( \text{NOT} \ X_{14} \) - \( P_{52} \) trigger event
\( X_7 = \text{False} \) - Condition variable
Expected Output: \( \text{PITCH} \)

329) Test specification APPR-VS-4

Prefix: \( X_{12} \) - Reach \( \text{VS} \)
Test case value: \( X_8 = \text{True} \) - \( P_{53} \) trigger event
\( X_{14} \) - \( P_{52} \) trigger event
\( X_7 = \text{True} \) - Condition variable
Expected Output: \( \text{FLC} \)

330) Test specification APPR-VS-5

Prefix: \( X_{12} \) - Reach \( \text{VS} \)
Test case value: \( X_8 = \text{True} \) - \( P_{53} \) trigger event
X14 - P55 trigger event
X7 = True - Condition variable

Expected Output: GA

331) Test specification APPR-ALTSEL-1

Prefix: X3 = True - Reach ALTSEL
Test case value: X8 = True - P53 trigger event
X3 = True - P43 trigger event

Expected Output: ALTSEL

332) Test specification APPR-ALTSEL-2

Prefix: X3 = True - Reach ALTSEL
Test case value: X8 = True - P53 trigger event
X8 = True - P52 trigger event
NOT X14 - P52 trigger event
X7 = True - Condition variable

Expected Output: FLC

333) Test specification APPR-ALTSEL-3

Prefix: X3 = True - Reach ALTSEL
Test case value: X8 = True - P53 trigger event
X8 = True - P52 trigger event
NOT X14 - P52 trigger event
X7 = False - Condition variable

Expected Output: PITCH

334) Test specification APPR-ALTSEL-4

Prefix: X3 = True - Reach ALTSEL
Test case value: X8 = True - P53 trigger event
X14 - P52 trigger event
X7 = True - Condition variable

Expected Output: FLC

335) Test specification APPR-ALTSEL-5

Prefix: X3 = True - Reach ALTSEL
Test case value: X8 = True - P53 trigger event
X14 - P55 trigger event
X7 = True - Condition variable

Expected Output: GA

336) Test specification APPR-ALTHOLD-1

Prefix: X11 - Reach ALTHOLD
Test case value: X8 = True - P53 trigger event
X3 = True - P43 trigger event

Expected Output: ALTSEL
337) Test specification APPR-ALTHOLD-2

Prefix: X11
Test case value: X8 = True
X8 = True
X7 = True
Expected Output: FLC

338) Test specification APPR-ALTHOLD-3

Prefix: X11
Test case value: X8 = True
X8 = True
X7 = False
Expected Output: PITCH

339) Test specification APPR-ALTHOLD-4

Prefix: X11
Test case value: X8 = True
X4
X7 = True
Expected Output: FLC

340) Test specification APPR-ALTHOLD-5

Prefix: X11
Test case value: X8 = True
X4
X7 = True
Expected Output: GA

341) Test specification APPR-PITCH-1

Prefix: X2
Test case value: X8 = True
X3 = True
Expected Output: ALTSEL

342) Test specification APPR-PITCH-2

Prefix: X2
Test case value: X8 = True
X8 = True
X7 = True
Expected Output: FLC

343) Test specification APPR-PITCH-3
Prefix: X2 - Reach PITCH
Test case value: X8 = True - P53 trigger event
X8 = True - P52 trigger event
NOT X14 - P52 trigger event
X7 = False - Condition variable
Expected Output: PITCH

344) Test specification APPR-PITCH-4

Prefix: X2 - Reach PITCH
Test case value: X8 = True - P53 trigger event
X14 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

345) Test specification APPR-PITCH-5

Prefix: X2 - Reach PITCH
Test case value: X8 = True - P53 trigger event
X14 - P55 trigger event
X7 = True - Condition variable
Expected Output: GA

346) Test specification APPR-FLC-1

Prefix: X13 - Reach FLC
Test case value: X8 = True - P53 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

347) Test specification APPR-FLC-2

Prefix: X13 - Reach FLC
Test case value: X8 = True - P53 trigger event
X8 = True - P52 trigger event
NOT X14 - P52 trigger event
X7 = True - Condition variable
Expected Output: FLC

348) Test specification APPR-FLC-3

Prefix: X13 - Reach FLC
Test case value: X8 = True - P53 trigger event
X8 = True - P52 trigger event
NOT X14 - P52 trigger event
X7 = False - Condition variable
Expected Output: PITCH

349) Test specification APPR-FLC-4

Prefix: X13 - Reach FLC
Test case value: X8 = True
X14
X7 = True
- Condition variable
Expected Output: FLC

350) Test specification APPR-FLC-5

Prefix: X13
Test case value: X8 = True
X14
X7 = True
- Condition variable
Expected Output: GA

351) Test specification GA-VS-1

Prefix: X12
Test case value: X14
X1 = True
- P41 trigger event
Expected Output: PITCH

352) Test specification GA-VS-2

Prefix: X12
Test case value: X14
X2
- P42 trigger event
Expected Output: PITCH

353) Test specification GA-VS-3

Prefix: X12
Test case value: X14
X3 = True
- P43 trigger event
Expected Output: ALTSEL

354) Test specification GA-VS-4

Prefix: X12
Test case value: X14
X11
- P46 trigger event
Expected Output: ALTHOLD

355) Test specification GA-VS-5

Prefix: X12
Test case value: X14
X12
- P48 trigger event
Expected Output: VS

356) Test specification GA-VS-6

Prefix: X12
- Reach VS
Test case value: X14 - P55 trigger event
X13 - P50 trigger event
Expected Output: FLC

357) Test specification GA-VS-7

Prefix: X12 - Reach VS
Test case value: X14 - P55 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

358) Test specification GA-VS-8

Prefix: X12 - Reach VS
Test case value: X14 - P55 trigger event
X9 = True - P56 trigger event
Expected Output: PITCH

359) Test specification GA-VS-9

Prefix: X12 - Reach VS
Test case value: X14 - P55 trigger event
X10 = False - P57 trigger event
Expected Output: PITCH

360) Test specification GA-ALTSEL-1

Prefix: X3 = True - Reach ALTSEL
Test case value: X14 - P55 trigger event
X1 = True - P41 trigger event
Expected Output: PITCH

361) Test specification GA-ALTSEL-2

Prefix: X3 = True - Reach ALTSEL
Test case value: X14 - P55 trigger event
X2 - P42 trigger event
Expected Output: PITCH

362) Test specification GA-ALTSEL-3

Prefix: X3 = True - Reach ALTSEL
Test case value: X14 = True - P55 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

363) Test specification GA-ALTSEL-4

Prefix: X3 = True - Reach ALTSEL
Test case value: X14 - P55 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

364) Test specification GA-ALTSEL-5

Prefix: \(X_3 = \text{True}\) - Reach ALTSEL
Test case value: \(X_{14}\) - P55 trigger event
\(X_{12}\) - P48 trigger event
Expected Output: VS

365) Test specification GA-ALTSEL-6

Prefix: \(X_3 = \text{True}\) - Reach ALTSEL
Test case value: \(X_{14}\) - P55 trigger event
\(X_{13}\) - P50 trigger event
Expected Output: FLC

366) Test specification GA-ALTSEL-7

Prefix: \(X_3 = \text{True}\) - Reach ALTSEL
Test case value: \(X_{14}\) - P55 trigger event
\(X_8 = \text{True}\) - P53 trigger event
Expected Output: APPR

367) Test specification GA-ALTSEL-8

Prefix: \(X_3 = \text{True}\) - Reach ALTSEL
Test case value: \(X_{14}\) - P55 trigger event
\(X_9 = \text{True}\) - P56 trigger event
Expected Output: PITCH

368) Test specification GA-ALTSEL-9

Prefix: \(X_3 = \text{True}\) - Reach ALTSEL
Test case value: \(X_{14}\) - P55 trigger event
\(X_{10} = \text{False}\) - P57 trigger event
Expected Output: PITCH

369) Test specification GA-ALTHOLD-1

Prefix: \(X_{11}\) - Reach ALTHOLD
Test case value: \(X_{14}\) - P55 trigger event
\(X_1 = \text{True}\) - P41 trigger event
Expected Output: PITCH

370) Test specification GA-ALTHOLD-2

Prefix: \(X_{11}\) - Reach ALTHOLD
Test case value: \(X_{14}\) - P55 trigger event
\(X_2\) - P42 trigger event
Expected Output: PITCH
371) Test specification GA-ALTHOLD-3

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P55 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

372) Test specification GA-ALTHOLD-4

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P55 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

373) Test specification GA-ALTHOLD-5

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P55 trigger event
X12 - P48 trigger event
Expected Output: VS

374) Test specification GA-ALTHOLD-6

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P55 trigger event
X13 - P50 trigger event
Expected Output: FLC

375) Test specification GA-ALTHOLD-7

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P55 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

376) Test specification GA-ALTHOLD-8

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P55 trigger event
X9 = True - P56 trigger event
Expected Output: PITCH

377) Test specification GA-ALTHOLD-9

Prefix: X11 - Reach ALTHOLD
Test case value: X14 - P55 trigger event
X10 = False - P57 trigger event
Expected Output: PITCH

378) Test specification GA-PITCH-1
Prefix: X2 - Reach PITCH
Test case value: X14 - P55 trigger event
X1 = True - P41 trigger event
Expected Output: PITCH

379) Test specification GA-PITCH-2
Prefix: X2 - Reach PITCH
Test case value: X14 - P55 trigger event
X2 - P42 trigger event
Expected Output: PITCH

380) Test specification GA-PITCH-3
Prefix: X2 - Reach PITCH
Test case value: X14 - P55 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

381) Test specification GA-PITCH-4
Prefix: X2 - Reach PITCH
Test case value: X14 - P55 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

382) Test specification GA-PITCH-5
Prefix: X2 - Reach PITCH
Test case value: X14 - P55 trigger event
X12 - P48 trigger event
Expected Output: VS

383) Test specification GA-PITCH-6
Prefix: X2 - Reach PITCH
Test case value: X14 - P55 trigger event
X13 - P50 trigger event
Expected Output: FLC

384) Test specification GA-PITCH-7
Prefix: X2 - Reach PITCH
Test case value: X14 - P55 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

385) Test specification GA-PITCH-8
Prefix: X2 - Reach PITCH
Test case value: X14 - P55 trigger event
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X9 = True</td>
<td>Prefix: X8 = True</td>
<td>Prefix: X8 = True</td>
<td>Prefix: X8 = True</td>
<td>Prefix: X8 = True</td>
<td>Prefix: X8 = True</td>
<td>Prefix: X8 = True</td>
</tr>
<tr>
<td>P56 trigger event</td>
<td>- Reach APPR</td>
<td>- Reach APPR</td>
<td>- Reach APPR</td>
<td>- Reach APPR</td>
<td>- Reach APPR</td>
<td>- Reach APPR</td>
</tr>
<tr>
<td>Test case value: X14</td>
<td>Test case value: X14</td>
<td>Test case value: X14</td>
<td>Test case value: X14</td>
<td>Test case value: X14</td>
<td>Test case value: X14</td>
<td>Test case value: X14</td>
</tr>
<tr>
<td>X10 = False</td>
<td>X3 = True</td>
<td>X2</td>
<td>X3 = True</td>
<td>X11</td>
<td>X12</td>
<td>X13</td>
</tr>
<tr>
<td>P55 trigger event</td>
<td>- P42 trigger event</td>
<td>- P41 trigger event</td>
<td>- P43 trigger event</td>
<td>- P46 trigger event</td>
<td>- P48 trigger event</td>
<td>- P50 trigger event</td>
</tr>
</tbody>
</table>
393) Test specification GA-APPR-7

Prefix: X8 = True - Reach APPR
Test case value: X14 - P55 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

394) Test specification GA-APPR-8

Prefix: X8 = True - Reach APPR
Test case value: X14 - P55 trigger event
X9 = True - P56 trigger event
Expected Output: PITCH

395) Test specification GA-APPR-9

Prefix: X8 = True - Reach APPR
Test case value: X14 - P55 trigger event
X10 = False - P57 trigger event
Expected Output: PITCH

396) Test specification GA-FLC-1

Prefix: X13 - Reach FLC
Test case value: X14 - P55 trigger event
X1 = True - P41 trigger event
Expected Output: PITCH

397) Test specification GA-FLC-2

Prefix: X13 - Reach FLC
Test case value: X14 - P55 trigger event
X2 - P42 trigger event
Expected Output: PITCH

398) Test specification GA-FLC-3

Prefix: X13 - Reach FLC
Test case value: X14 - P55 trigger event
X3 = True - P43 trigger event
Expected Output: ALTSEL

399) Test specification GA-FLC-4

Prefix: X13 - Reach FLC
Test case value: X14 - P55 trigger event
X11 - P46 trigger event
Expected Output: ALTHOLD

400) Test specification GA-FLC-5
Prefix: X13 - Reach FLC
Test case value: X14 - P55 trigger event
X12 - P48 trigger event
Expected Output: VS

401) Test specification GA-FLC-6

Prefix: X13 - Reach FLC
Test case value: X14 - P55 trigger event
X13 - P50 trigger event
Expected Output: FLC

402) Test specification GA-FLC-7

Prefix: X13 - Reach FLC
Test case value: X14 - P55 trigger event
X8 = True - P53 trigger event
Expected Output: APPR

403) Test specification GA-FLC-8

Prefix: X13 - Reach FLC
Test case value: X14 - P55 trigger event
X9 = True - P56 trigger event
Expected Output: PITCH

404) Test specification GA-FLC-9

Prefix: X13 - Reach FLC
Test case value: X14 - P55 trigger event
X10 = False - P57 trigger event
Expected Output: PITCH

6.12 Flight Level Change Submode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>58</td>
<td>-</td>
<td>@T(Mode_Active_Vertical = FLC) AND NOT @T(term_Overspeed)</td>
<td>Track</td>
</tr>
<tr>
<td>59</td>
<td>Overspeed</td>
<td>@F(term_Overspeed)</td>
<td>Track</td>
</tr>
<tr>
<td>60</td>
<td>-</td>
<td>@T(Mode_Active_Vertical = FLC) AND @T(term_Overspeed)</td>
<td>Overspeed</td>
</tr>
<tr>
<td>61</td>
<td>Track</td>
<td>@T(term_Overspeed)</td>
<td>Overspeed</td>
</tr>
</tbody>
</table>

Table 6.12: Flight Level Change Submode Transition Table
(Table A.12, pg. 81 in the FGS report)

Definitions:
- X1 = (mode_Active_Vertical + FLC)
- X2 = @term_Overspeed
- X1' -- After-value of trigger event X1
- $X_2'$ -- After-value of trigger event $X_2$
- $\text{Pre-P58} = @(\text{FLC Switched_Pressed})$
- $\text{Pre-P60} = @(\text{FLC Switched_Pressed})$

### 6.12.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_1'$</th>
<th>$X_2'$</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P58 Entry to FLC</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>Track</td>
</tr>
<tr>
<td>Entry to FLC</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>Entry to FLC</td>
</tr>
<tr>
<td>Entry to FLC</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>Entry to FLC</td>
</tr>
<tr>
<td>Entry to FLC</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>Entry to FLC</td>
</tr>
<tr>
<td>Entry to FLC</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>Entry to FLC</td>
</tr>
<tr>
<td>P59 Overspeed</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>F</td>
<td>Track</td>
</tr>
<tr>
<td>Overspeed</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>F</td>
<td>Overspeed</td>
</tr>
<tr>
<td>Overspeed</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>T</td>
<td>Overspeed</td>
</tr>
<tr>
<td>P60 Entry to FLC</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>Overspeed</td>
</tr>
<tr>
<td>Entry to FLC</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>Entry to FLC</td>
</tr>
<tr>
<td>Entry to FLC</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>Entry to FLC</td>
</tr>
<tr>
<td>Entry to FLC</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>Entry to FLC</td>
</tr>
<tr>
<td>Entry to FLC</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>Entry to FLC</td>
</tr>
<tr>
<td>P61 Track</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>T</td>
<td>Overspeed</td>
</tr>
<tr>
<td>Track</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>T</td>
<td>Track</td>
</tr>
<tr>
<td>Track</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>F</td>
<td>Track</td>
</tr>
</tbody>
</table>

### Test specifications:

1) Test specification P58-1:

- **Prefix:** Pre-P58
- **Test case value:**
  - $X_1$ = False
  - $X_2$ = True
  - $X_1'$ = True
  - $X_2'$ = False
- **Expected Output:** Track

2) Test specification P58-2:

- **Prefix:** Pre-P58
- **Test case value:**
  - $X_1$ = True
  - $X_2$ = True
  - $X_1'$ = True
  - $X_2'$ = False
- **Expected Output:** Entry to FLC

3) Test specification P58-3:

- **Prefix:** Pre-P58
- **Test case value:**
  - $X_1$ = True
  - $X_2$ = True
  - $X_1'$ = True
  - $X_2'$ = False
- **Expected Output:** Entry to FLC
<table>
<thead>
<tr>
<th>Test case value:</th>
<th>- Trigger before-value</th>
<th>- Trigger before-value</th>
<th>- Trigger event</th>
<th>- Trigger event</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1 = False</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X2 = False</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X1 = True</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X2 = False</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Expected Output: Entry to FLC

4) Test specification P58-4:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>- Reach Entry to FLC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-P58</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test case value:</th>
<th>- Trigger before-value</th>
<th>- Trigger before-value</th>
<th>- Trigger event</th>
<th>- Trigger event</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1 = False</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X2 = True</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X1 = False</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X2 = False</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Expected Output: Entry to FLC

5) Test specification P58-5:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>- Reach Entry to FLC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-P58</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test case value:</th>
<th>- Trigger before-value</th>
<th>- Trigger before-value</th>
<th>- Trigger event</th>
<th>- Trigger event</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1 = False</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X2 = True</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X1 = True</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X2 = True</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Expected Output: Entry to FLC

6) Test specification P59-1:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>- Reach Overspeed</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1 = True</td>
<td></td>
</tr>
<tr>
<td>X2 = True</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test case value:</th>
<th>- Trigger before-value</th>
<th>- Trigger event</th>
</tr>
</thead>
<tbody>
<tr>
<td>X2 = False</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X2 = False</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Expected Output: Track

7) Test specification P59-2:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>- Reach Overspeed</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1 = True</td>
<td></td>
</tr>
<tr>
<td>X2 = True</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test case value:</th>
<th>- Trigger before-value</th>
<th>- Trigger event</th>
</tr>
</thead>
<tbody>
<tr>
<td>X2 = False</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X2 = False</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Expected Output: Overspeed

8) Test specification P59-3:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>- Reach Overspeed</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1 = True</td>
<td></td>
</tr>
<tr>
<td>X2 = True</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test case value:</th>
<th>- Trigger before-value</th>
<th>- Trigger event</th>
</tr>
</thead>
<tbody>
<tr>
<td>X2 = False</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X2 = False</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Expected Output: Overspeed

9) Test specification P60-1:

<table>
<thead>
<tr>
<th>Prefix:</th>
<th>- Reach Entry to FLC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-P60</td>
<td></td>
</tr>
</tbody>
</table>

186
Test case value:  X1 = False  - Trigger before-value  
X2 = False  - Trigger before-value  
X1 = True  - Trigger event  
X2 = True  - Trigger event  
Expected Output:  Overspeed  

10) Test specification P60-2:  

Prefix:  Pre-P60  - Reach Entry to FLC  
Test case value:  X1 = True  - Trigger before-value  
X2 = False  - Trigger before-value  
X1 = True  - Trigger event  
X2 = True  - Trigger event  
Expected Output:  Entry to FLC  

11) Test specification P60-3:  

Prefix:  Pre-P60  - Reach Entry to FLC  
Test case value:  X1 = False  - Trigger before-value  
X2 = True  - Trigger before-value  
X1 = True  - Trigger event  
X2 = True  - Trigger event  
Expected Output:  Entry to FLC  

12) Test specification P60-4:  

Prefix:  Pre-P60  - Reach Entry to FLC  
Test case value:  X1 = False  - Trigger before-value  
X2 = False  - Trigger before-value  
X1 = False  - Trigger event  
X2 = True  - Trigger event  
Expected Output:  Entry to FLC  

13) Test specification P60-5:  

Prefix:  Pre-P60  - Reach Entry to FLC  
Test case value:  X1 = False  - Trigger before-value  
X2 = False  - Trigger before-value  
X1 = True  - Trigger event  
X2 = False  - Trigger event  
Expected Output:  Entry to FLC  

14) Test specification P61-1:  

Prefix:  X1 = True  - Reach Track  
X2 = False  
Test case value:  X2 = False  - Trigger before-value  
X2 = True  - Trigger event  
Expected Output:  Overspeed  

15) Test specification P61-2:
Prefix: \( X_1 = \text{True} \) - Reach Track  \( X_2 = \text{False} \)
Test case value: \( X_2 = \text{True} \) - Trigger before-value  \( X_2 = \text{True} \) - Trigger event
Expected Output: Track

16) Test specification P61-3:
Prefix: \( X_1 = \text{True} \) - Reach Track  \( X_2 = \text{False} \)
Test case value: \( X_2 = \text{False} \) - Trigger before-value  \( X_2 = \text{False} \) - Trigger event
Expected Output: Track

6.12.2 Transition Pair Coverage Level Requirements:
The pairs for the Track Mode are:

\((P58 \text{ or } P59) : P61\)

The pairs for the Overspeed Mode are:

\((P60 \text{ or } P61) : P59\)

\[
\begin{array}{ccc}
X_1 & X_2 \\
\text{Track} & \text{Overspeed} & - & F & \text{Track} \\
\text{OR} & \text{Entry to FLC} & T & F & \text{Track} \\
\text{Track} & - & T & \text{Overspeed} \\
\text{Overspeed} & \text{Track} & - & T & \text{Overspeed} \\
\text{OR} & \text{Entry to FLC} & T & T & \text{Overspeed} \\
\text{Overspeed} & - & F & \text{Track} \\
\end{array}
\]

Test specifications

1) Test specification Track:
Prefix: \( X_1 = \text{True} \) - Reach Overspeed  \( X_2 = \text{True} \)
Test case value: \( X_2 = \text{False} \) - P59 trigger event  \( X_2 = \text{False} \) - P58 trigger event  \( X_2 = \text{True} \) - P61 trigger event
Expected Output: Overspeed

2) Test specification Overspeed:

Prefix:  
- X1 = True  
- X2 = False

Test case value:
- X1 = True  
- X2 = True  
- X2 = False  
- X2 = True

Expected Output: Track

6.13 Altitude Select Mode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>62</td>
<td>CLEARED</td>
<td>@F{(Mode_Active_Vertical ∈ {APPR, GA, ALTHOLD})}</td>
<td>ENABLED</td>
</tr>
<tr>
<td>63</td>
<td>ENABLED</td>
<td>@T{(Mode_Active_Vertical ∈ {APPR, GA, ALTHOLD})}</td>
<td>CLEARED</td>
</tr>
</tbody>
</table>

Table 6.13: Altitude Select Mode Transition Table  
(Table A.13, pg. 84 in the FGS report)

Definitions:
- X1 = mode_Active_Vertical ∈ APPR, GA, ALTHOLD
- X1' = After-value of trigger event X1

6.13.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre</th>
<th>X1</th>
<th>X1'</th>
<th>Post</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>State</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P62</td>
<td>CLEARED</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>CLEARED</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>CLEARED</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>P63</td>
<td>ENABLED</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>ENABLED</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>ENABLED</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

Test specifications:

1) Test specification P62-1:

Prefix:  
- X1 = True  
- Reach CLEARED

Test case value:
- X1 = True  
- Trigger before-value  
- X1 = False  
- Trigger event

Expected Output: ENABLED

2) Test specification P62-2:

Prefix:  
- X1 = True  
- Reach CLEARED
Test case value: $X_1 = \text{False}$ - Trigger before-value
$X_1 = \text{False}$ - Trigger event
Expected Output: CLEARED

3) Test specification P62-3:
Prefix: $X_1 = \text{True}$ - Reach CLEARED
Test case value: $X_1 = \text{True}$ - Trigger before-value
$X_1 = \text{True}$ - Trigger event
Expected Output: CLEARED

4) Test specification P63-1:
Prefix: $X_1 = \text{False}$ - Reach ENABLED
Test case value: $X_1 = \text{False}$ - Trigger before-value
$X_1 = \text{True}$ - Trigger event
Expected Output: CLEARED

5) Test specification P63-2:
Prefix: $X_1 = \text{False}$ - Reach ENABLED
Test case value: $X_1 = \text{True}$ - Trigger before-value
$X_1 = \text{True}$ - Trigger event
Expected Output: ENABLED

6) Test specification P63-3:
Prefix: $X_1 = \text{False}$ - Reach ENABLED
Test case value: $X_1 = \text{False}$ - Trigger before-value
$X_1 = \text{False}$ - Trigger event
Expected Output: ENABLED

6.13.2 Transition Pair Coverage Level Requirements:
The pairs for the CLEARED Mode are:

P63 : P62

The pairs for the ENABLED Mode are:

P62 : P63

$X_1$

<table>
<thead>
<tr>
<th></th>
<th>ENABLED</th>
<th></th>
<th>CLEARED</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLEARED</td>
<td></td>
<td>T</td>
<td>CLEARED</td>
</tr>
<tr>
<td>CLEARED</td>
<td></td>
<td>F</td>
<td>ENABLED</td>
</tr>
<tr>
<td>ENABLED</td>
<td></td>
<td>F</td>
<td>ENABLED</td>
</tr>
<tr>
<td>ENABLED</td>
<td></td>
<td>T</td>
<td>CLEARED</td>
</tr>
</tbody>
</table>

Test specifications
1) Test specification CLEARED:

Prefix: \( X_1 = False \)  
Test case value: \( X_1 = True \) \( \rightarrow \) Reach ENABLED  
\( X_1 = False \) \( \rightarrow \) P62 trigger event  
\( X_1 = True \) \( \rightarrow \) P63 trigger event  
Expected Output: ENABLED

2) Test specification ENABLED:

Prefix: \( X_1 = True \)  
Test case value: \( X_1 = False \) \( \rightarrow \) P62 trigger event  
\( X_1 = True \) \( \rightarrow \) P63 trigger event  
Expected Output: CLEARED

6.14 Altitude Select ENABLED Submode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>ARMED</td>
<td>( @T(\text{term}<em>{\text{ALTSEL},\text{Cond}} = \text{Capture} ) ) ( \text{AND} ) ( \text{Duration(INMODE)} &gt; \text{const}</em>{\text{min,armed,period}} )</td>
<td>ACTIVE</td>
</tr>
<tr>
<td>65</td>
<td>ACTIVE</td>
<td>( @T(\text{Mode}_{\text{Active,Vertical}} \in {\text{APPR, GA, ALTSEL, ALTHOLD}}) )</td>
<td>ARMED</td>
</tr>
</tbody>
</table>

Table 6.14: Altitude Select ENABLED Submode Transition Table  
(Table A.14, pg. 84 in the FGS report)

Definitions:

- \( X_1 = (\text{term}_{\text{ALTSEL,Cond}} = \text{Capture}) \)
- \( X_2 = (\text{Duration(INMODE)} > \text{const}_{\text{min,armed,period}}) \)
- \( X_3 = (\text{mode}_{\text{Active,Vertical}} \in \{\text{APPR, GA, ALTSEL, ALTHOLD}\}) \)
- \( X_1' \rightarrow \) After-value of trigger event \( X_1 \)
- \( X_2' \rightarrow \) After-value of trigger event \( X_2 \)
- \( X_3' \rightarrow \) After-value of trigger event \( X_3 \)

6.14.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X1'</th>
<th>X2'</th>
<th>X3'</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P64</td>
<td>ARMED</td>
<td>F</td>
<td>F</td>
<td>-</td>
<td>T</td>
<td>T</td>
<td>-</td>
</tr>
<tr>
<td>ARMED</td>
<td>T</td>
<td>T</td>
<td>-</td>
<td>T</td>
<td>T</td>
<td>-</td>
<td>ARMED</td>
</tr>
<tr>
<td>ARMED</td>
<td>T</td>
<td>T</td>
<td>-</td>
<td>F</td>
<td>T</td>
<td>-</td>
<td>ARMED</td>
</tr>
<tr>
<td>ARMED</td>
<td>T</td>
<td>T</td>
<td>-</td>
<td>T</td>
<td>F</td>
<td>-</td>
<td>ARMED</td>
</tr>
<tr>
<td>P65</td>
<td>ACTIVE</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>T</td>
</tr>
<tr>
<td>ACTIVE</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>-</td>
<td>-</td>
<td>T</td>
<td>ACTIVE</td>
</tr>
<tr>
<td>ACTIVE</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>-</td>
<td>-</td>
<td>F</td>
<td>ACTIVE</td>
</tr>
</tbody>
</table>

Test specifications:
1) Test specification P64-1:

Prefix: \( X_3 = \text{True} \) - Reach ARMED
Test case value: \( X_1 = \text{False} \) - Trigger before-value
\( X_2 = \text{False} \) - Trigger before-value
\( X_1 = \text{True} \) - Trigger event
\( X_2 = \text{True} \) - Trigger event
Expected Output: ACTIVE

2) Test specification P64-2:

Prefix: \( X_3 = \text{True} \) - Reach ARMED
Test case value: \( X_1 = \text{True} \) - Trigger before-value
\( X_2 = \text{True} \) - Trigger before-value
\( X_1 = \text{True} \) - Trigger event
\( X_2 = \text{True} \) - Trigger event
Expected Output: ARMED

3) Test specification P64-3:

Prefix: \( X_3 = \text{True} \) - Reach ARMED
Test case value: \( X_1 = \text{True} \) - Trigger before-value
\( X_2 = \text{True} \) - Trigger before-value
\( X_1 = \text{False} \) - Trigger event
\( X_2 = \text{True} \) - Trigger event
Expected Output: ARMED

4) Test specification P64-4:

Prefix: \( X_3 = \text{True} \) - Reach ARMED
Test case value: \( X_1 = \text{True} \) - Trigger before-value
\( X_2 = \text{True} \) - Trigger before-value
\( X_1 = \text{True} \) - Trigger event
\( X_2 = \text{False} \) - Trigger event
Expected Output: ARMED

5) Test specification P65-1:

Prefix: \( X_1 = \text{True} \) - Reach ACTIVE
\( X_2 = \text{True} \)
Test case value: \( X_3 = \text{False} \) - Trigger before-value
\( X_3 = \text{True} \) - Trigger event
Expected Output: ARMED

6) Test specification P65-2:

Prefix: \( X_1 = \text{True} \) - Reach ACTIVE
\( X_2 = \text{True} \)
Test case value: \( X_3 = \text{True} \) - Trigger before-value
\( X_3 = \text{True} \) - Trigger event
Expected Output: ACTIVE
7) Test specification P65-3:

Prefix: \( X_1 = \text{True} \) - Reach ACTIVE
\( X_2 = \text{True} \)

Test case value: \( X_3 = \text{False} \) - Trigger before-value
\( X_2 = \text{True} \)
\( X_3 = \text{False} \) - Trigger event

Expected Output: ACTIVE

6.14.2 Transition Pair Coverage Level Requirements:

The pairs for the ACTIVE Mode are:

\[ P64 : P65 \]

The pairs for the ARMED Mode are:

\[ P65 : P64 \]

\[
\begin{array}{ccc}
X_1 & X_2 & X_3 \\
\hline
\text{ACTIVE} & \text{ARMED} & T & T & - & \text{ACTIVE} \\
& \text{ACTIVE} & - & - & T & \text{ARMED} \\
\text{ARMED} & \text{ACTIVE} & - & - & T & \text{ARMED} \\
& \text{ARMED} & T & T & - & \text{ACTIVE} \\
\end{array}
\]

Test specifications

1) Test specification ACTIVE:

Prefix: \( X_3 = \text{True} \) - Reach ARMED

Test case value: \( X_1 = \text{True} \) - P64 trigger event
\( X_2 = \text{True} \)
\( X_3 = \text{True} \) - P65 trigger event

Expected Output: ACTIVE

2) Test specification ARMED:

Prefix: \( X_1 = \text{True} \) - Reach ACTIVE
\( X_2 = \text{True} \)

Test case value: \( X_3 = \text{True} \) - P65 trigger event
\( X_1 = \text{True} \) - P64 trigger event
\( X_2 = \text{True} \)

Expected Output: ARMED

6.15 Altitude Select ACTIVE Submode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>66</td>
<td>Capture</td>
<td>( \text{@T(\text{term_ALTSEI_Cond = Track \ AND Duration(INMODE) &gt; const_min_armed_period)}} )</td>
<td>Track</td>
</tr>
</tbody>
</table>
Table 6.15: Altitude Select ACTIVE Submode Transition Table

(Table A.15, pg. 84 in the FGS report)

Definitions:

- $X_1 = (\text{term}_{\text{ALTSEL}}\_\text{Cond} = \text{Track})$
- $X_2 = (\text{Duration}(<INMODE>) > \text{const}_{\text{min}_{\text{armed}}\_\text{period}})$
- $X_1' = \text{After-value of trigger event } X_1$
- $X_2' = \text{After-value of trigger event } X_2$
- $\text{Pre-P66-1} = (\text{term}_{\text{ALTSEL}}\_\text{Cond} = \text{Capture})$
- $\text{Pre-P66-2} = (\text{Duration}(<INMODE>) > \text{const}_{\text{min}_{\text{armed}}\_\text{period}})$

6.15.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>X1</th>
<th>X2</th>
<th>X1'</th>
<th>X2'</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>P66 Capture</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>Track</td>
</tr>
<tr>
<td>Capture</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>Capture</td>
</tr>
<tr>
<td>Capture</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>Capture</td>
</tr>
<tr>
<td>Capture</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>Capture</td>
</tr>
</tbody>
</table>

Test specifications:

1) Test specification P66-1:

Prefix: $\text{Pre-P66-1} = \text{True}$ - Reach Entry to Capture
$\text{Pre-P66-2} = \text{True}$

Test case value: $X_1 = \text{False}$ - Trigger before-value
$X_2 = \text{False}$ - Trigger before-value
$X_1 = \text{True}$ - Trigger event
$X_2 = \text{True}$ - Trigger event

Expected Output: Track

2) Test specification P66-2:

Prefix: $\text{Pre-P66-1} = \text{True}$ - Reach Entry to Capture
$\text{Pre-P66-2} = \text{True}$

Test case value: $X_1 = \text{True}$ - Trigger before-value
$X_2 = \text{True}$ - Trigger before-value
$X_1 = \text{True}$ - Trigger event
$X_2 = \text{True}$ - Trigger event

Expected Output: Capture

3) Test specification P66-3:

Prefix: $\text{Pre-P66-1} = \text{True}$ - Reach Entry to Capture
Pre-P66-2 = True

Test case value:  
- X1 = True  - Trigger before-value
- X2 = True  - Trigger before-value
- X1 = False  - Trigger event
- X2 = True  - Trigger event

Expected Output: Capture

4) Test specification P66-4:

Prefix:  
- Pre-P66-1 = True  - Reach Entry to Capture
- Pre-P66-2 = True

Test case value:  
- X1 = True  - Trigger before-value
- X2 = True  - Trigger before-value
- X1 = True  - Trigger event
- X2 = False  - Trigger event

Expected Output: Capture

6.15.2 Transition Pair Coverage Level Requirements:

NONE.

6.16 Vertical Approach Mode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>67</td>
<td>CLEARED</td>
<td>@(mode_Active_Lateral = APPR/Track)</td>
<td>ENABLED</td>
</tr>
<tr>
<td>68</td>
<td>ENABLED</td>
<td>@F(mode_Active_Lateral = APPR/Track)</td>
<td>CLEARED</td>
</tr>
</tbody>
</table>

Table 6.16: Vertical Approach Mode Transition Table
(Table A.16, pg. 87 in the FGS report)

Definitions:

- X1 = (mode_Active_Lateral = APPR/Track)
- X1’ = After-value of trigger event X1

6.16.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre</th>
<th>From</th>
<th>X1</th>
<th>X1’</th>
<th>Post</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td></td>
<td></td>
<td></td>
<td>State</td>
</tr>
<tr>
<td>P67</td>
<td>CLEARED</td>
<td>F</td>
<td>T</td>
<td>ENABLED</td>
</tr>
<tr>
<td></td>
<td>CLEARED</td>
<td>T</td>
<td>T</td>
<td>CLEARED</td>
</tr>
<tr>
<td></td>
<td>CLEARED</td>
<td>F</td>
<td>F</td>
<td>CLEARED</td>
</tr>
<tr>
<td>P68</td>
<td>ENABLED</td>
<td>T</td>
<td>F</td>
<td>CLEARED</td>
</tr>
<tr>
<td></td>
<td>ENABLED</td>
<td>F</td>
<td>F</td>
<td>ENABLED</td>
</tr>
<tr>
<td></td>
<td>ENABLED</td>
<td>T</td>
<td>T</td>
<td>ENABLED</td>
</tr>
</tbody>
</table>

Test specifications:
1) Test specification P67-1:

Prefix: \( X_1 = \text{False} \) - Reach CLEARED
Test case value: \( X_1 = \text{False} \) - Trigger before-value
\( X_1 = \text{True} \) - Trigger event
Expected Output: ENABLED

2) Test specification P67-2:

Prefix: \( X_1 = \text{False} \) - Reach CLEARED
Test case value: \( X_1 = \text{True} \) - Trigger before-value
\( X_1 = \text{True} \) - Trigger event
Expected Output: ENABLED

3) Test specification P67-3:

Prefix: \( X_1 = \text{False} \) - Reach CLEARED
Test case value: \( X_1 = \text{False} \) - Trigger before-value
\( X_1 = \text{False} \) - Trigger event
Expected Output: ENABLED

4) Test specification P67-4:

Prefix: \( X_1 = \text{False} \) - Reach ENABLED
Test case value: \( X_1 = \text{False} \) - Trigger before-value
\( X_1 = \text{False} \) - Trigger event
Expected Output: CLEARED

5) Test specification P68-1:

Prefix: \( X_1 = \text{True} \) - Reach ENABLED
Test case value: \( X_1 = \text{True} \) - Trigger before-value
\( X_1 = \text{False} \) - Trigger event
Expected Output: CLEARED

6) Test specification P68-2:

Prefix: \( X_1 = \text{True} \) - Reach ENABLED
Test case value: \( X_1 = \text{False} \) - Trigger before-value
\( X_1 = \text{False} \) - Trigger event
Expected Output: ENABLED

7) Test specification P68-3:

Prefix: \( X_1 = \text{True} \) - Reach ENABLED
Test case value: \( X_1 = \text{True} \) - Trigger before-value
\( X_1 = \text{True} \) - Trigger event
Expected Output: ENABLED
6.16.2 Transition Pair Coverage Level Requirements:

The pairs for the ENABLED Mode are:

P67 : P68

The pairs for the CLEARED Mode are:

P68 : P67

X1

<table>
<thead>
<tr>
<th>ENABLED</th>
<th>CLEARED</th>
<th>T</th>
<th>ENABLED</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLED</td>
<td>F</td>
<td></td>
<td>CLEARED</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CLEARED</th>
<th>ENABLED</th>
<th>F</th>
<th>CLEARED</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLEARED</td>
<td>T</td>
<td></td>
<td>ENABLED</td>
</tr>
</tbody>
</table>

Test specifications

1) Test specification ENABLED:

Prefix: X1 = False - Reach CLEARED
Test case value: X1 = True  - P67 trigger event
X1 = False  - P68 trigger event
Expected Output: CLEARED

2) Test specification CLEARED:

Prefix: X1 = True - Reach ENABLED
Test case value: X1 = False  - P68 trigger event
X1 = True  - P67 trigger event
Expected Output: ENABLED

6.17 Vertical Approach ENABLED Submode Test Cases

<table>
<thead>
<tr>
<th>Id</th>
<th>From</th>
<th>Events</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>69</td>
<td>ARMED</td>
<td>@T(term_Vertical_Appr_Track_Cond_Met AND Duration(INMODE) &gt; const_minarmed_period)</td>
<td>Track</td>
</tr>
</tbody>
</table>

Table 6.17: Vertical Approach ENABLED Submode Transition Table (Table A.17, pg. 87 in the FGS report)

Definitions:

- X1 = (term_Vertical_APPR_Track_Cond_Met)
- X2 = (Duration(INMODE) > const_minarmed_period)
- X1' -- After-value of trigger event X1
- X2' -- After-value of trigger event X2
- Pre-P69 = (Mode_Active_Lateral = APPR/Track)
6.17.1 Full predicate coverage level test case requirements:

<table>
<thead>
<tr>
<th>Pre State</th>
<th>X1</th>
<th>X2</th>
<th>X1’</th>
<th>X2’</th>
<th>Post State</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARMED</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>Track</td>
</tr>
<tr>
<td>ARMED</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>ARMED</td>
</tr>
<tr>
<td>ARMED</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>ARMED</td>
</tr>
<tr>
<td>ARMED</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>ARMED</td>
</tr>
</tbody>
</table>

Test specifications:

1) Test specification P69-1:

Prefix: Pre-P69 = True - Reach ENABLED
Test case value:
X1 = False - Trigger before-value
X2 = False - Trigger before-value
X1 = True - Trigger event
X2 = True - Trigger event
Expected Output: Track

2) Test specification P69-2:

Prefix: Pre-P69 = True - Reach ENABLED
Test case value:
X1 = True - Trigger before-value
X2 = True - Trigger before-value
X1 = True - Trigger event
X2 = True - Trigger event
Expected Output: ARMED

3) Test specification P69-3:

Prefix: Pre-P69 = True - Reach ENABLED
Test case value:
X1 = True - Trigger before-value
X2 = True - Trigger before-value
X1 = False - Trigger event
X2 = True - Trigger event
Expected Output: ARMED

4) Test specification P69-4:

Prefix: Pre-P69 = True - Reach ENABLED
Test case value:
X1 = True - Trigger before-value
X2 = True - Trigger before-value
X1 = True - Trigger event
X2 = False - Trigger event
Expected Output: ARMED

6.17.2 Transition Pair Coverage Level Requirements:

NONE.
7 CONCLUSIONS

This report introduces a new technique for generating test data from formal software specifications. Formal specifications represent a significant opportunity for testing because they precisely describe the functionality of the software in a form that can be easily manipulated by automated means. This research addresses the problem of developing formalizable, measurable criteria for generating test cases from specifications. A model for generating tests from requirements/specifications and a derivation process for generating the test cases were presented. Results from applying the model and process to a small example were presented. This case study was evaluated using Atac to measure decision coverage, and the technique was found to achieve a high level of coverage. This result indicates that this technique can benefit software developers who construct formal specifications during development.

As an additional validation, tests were generated for specifications of an industrial software system, the Flight Guidance System. Construction of these tests resulted in several modifications to this technique, and found at least one problem with the specification.

One interesting result from the decision coverage is that only the functional specifications related to the cruise control state machine itself were covered. While this was certainly the focus of the study, several decisions having to do with the input were left out. For testing of real systems, the input specifications must be considered as well, either by adapting the method presented here, or by using another testing method.

The immediate goal of this research was to develop a model and formal criterion for generating tests from state-based specifications. Short term goals are to develop mechanical procedures to derive test cases from formal specifications, and apply the method to industrial software specifications supplied by the sponsor, Rockwell Collins, Inc. One observation from the case study and the industrial example is that it requires a lot of very detailed hand analysis to apply the technique. Both to save costs, and improve accuracy, a long term goal is to develop automated tool support to transform formal functional specifications into effective test cases. An eventual goal is to build an automatic test data generation tool for this technique.
8 ACKNOWLEDGMENTS

The efforts of Lei Sun in writing the cruise control software and generating most of the test cases are greatly appreciated. It is a pleasure to acknowledge Bellcore, and specifically Bob Horgan and Saul London, for the use of the testing tool ATAC and for support using it. Most importantly, of course, support from the sponsor, Rockwell Collins, Inc. is greatly appreciated.
References


A APPENDIX: CRUISE CONTROL IMPLEMENTATION

This program was written to model the cruise control specifications of Section 5. The test cases listed in that section were tested on this program.

/*////////////////////////////////////////////////////////////////////*/
/* Programmer : Lei Sun and Jeff Offutt */
/* Module : Cruise Control implementation. */
/* Purpose : To model the state diagram for the */
/* Usage : cruise [-s] [file] */
/* "-s" will allow the current state to be */
/* [file] is the input data file. If it's */
/* : Accepts sequences of VARIABLE VALUE pairs. */
/* : X t -- sets condition variable X TRUE. */
/* : X f -- sets condition variable X FALSE. */
/* : X T -- X=TRUE is a trigger event. */
/* : X F -- X=FALSE is a trigger event. */
/* Date : 8/97 */
/* Compile : On SITE, g++ -o cruise cruise.C */

#include <stdio.h>
#define FALSE 0
#define TRUE 1

/* enum StateType {OFF, INACTIVE, CRUISE, OVERRIDE, QUIT}; */

int CurState;
int Ignited, Running, TooFast, Brake, Activate, Deactivate, Resume;

/* Results from GetNextInput() .. exit from the program, */
/* set a condition variable, */
/* or a triggering event. */
#define EXIT -1
#define CONDITION 0
#define TRIGIGNITED 1
#define TRIGRUNNING 2
#define TRIGTOOFAST 3
#define TRIGBRAKE 4
#define TRIGACTIVATE 5
#define TRIGDEACTIVATE 6
#define TRIGRESUME 7

/* State */
#define OFF 1
#define INACTIVE 2
```c
#define CRUISE 3
#define OVERRIDE 4
#define QUIT 5

/* Defaults: */
int FileIn = FALSE, /* input is standard input */
    CurStPersists = TRUE; /* current state stays persistent */
FILE *FP, *fopen();

/* Programmer: Jeff Offutt */
/* Function: ParseArgs */
/* Purpose: To parse the arguments to set */
/*: global parameters. */
/* Date: 6/97 */

/***************************************************************************/
void ParseArgs (argc, argv)
int argc;
char *argv[];
{
    int argi;

    for (argi = 1; argi <= argc-1; argi++)
    {
        if (argv[argi][0] == '-')
        {
            if (argv[argi][1] == 's')
                CurStPersists = FALSE;
        }
        else
        {
            FileIn = TRUE;
            FP = fopen (argv[argi], "r");
            if (!FP)
            {
                /* File was not opened. */
                fprintf (stderr, "File %s is not available for reading.\n", argv[argi]);
                fprintf (stderr, "Running in interactive mode.\n");
                FileIn = FALSE;
            }
        }
    }
}*/

/***************************************************************************/
/* Programmer: Jeff Offutt */
/* Function: PrintState */
/* Purpose: Print a string representing the */
/*: state. */
/* Date: 6/97 */
/***************************************************************************/
void PrintState (S)
{
    switch (S)
    {
    case OFF:
        printf ("OFF");
        break;
    case INACTIVE:
        printf ("INACTIVE");
        break;
    case CRUISE:
        printf ("CRUISE");
        break;
    case OVERRIDE:
        printf ("OVERRIDE");
        break;
    case QUIT:
        printf ("QUIT");
        break;
    }
}

int GetState ()
{
    char state_str [80];
    if (FileIn == FALSE)
    {
        printf ("enter OFF, INACTIVE, CRUISE, or OVERRIDE: ");
        scanf ("%s", state_str);
    }
    else
        fscanf (FP, "%s", state_str);
    switch (state_str [0])
    {
    case 'O': case 'o':
        if (state_str[1] == 'F' || state_str[1] == 'f')
            return (OFF);
        else /* 'V' or 'v' */
            return (OVERRIDE);
        break;
    }
case 'I': case 'i':
    return (INACTIVE);
    break;
case 'C': case 'c':
    return (CRUISE);
    break;
default:
    return (OFF);
    break;
}

/**************************************************************************/
/* Programmer : Jeff Offutt */
/* Function : GetNextInput */
/* Purpose : Read next variable value pair. */
/* Date : 8/97 */
/**************************************************************************/
int GetNextInput()
{
    char variable [20];
    char c_value;
    int value;
    int ret_val = CONDITION;

    if (FileIn == FALSE)
    { /* Interactive mode */
        printf ("Input 'variable value' pair.\n");

        scanf ("%s", variable);
        scanf ("%c", &c_value);
        while (c_value == ' ') /* Skip spaces. */
            scanf ("%c", &c_value);
    }
    else
    {
        fscanf (FP, "%s", variable);
        fscanf (FP, "%c", &c_value);
        while (c_value == ' ') /* Skip spaces. */
            fscanf (FP, "%c", &c_value);
    }

    if (strcmp (variable, "Exit") == 0)
    {
        ret_val = EXIT;
        return (ret_val);
    }

    if ((c_value == 't') || (c_value == 'T'))
        value = TRUE;
else
    value = FALSE;

if (strcmp (variable, "Ignited") == 0)
{
    Ignited = value;
    if ((c_value == 'T') || (c_value == 'F'))
        ret_val = TRIGIGNITED;
}
else if (strcmp (variable, "Running") == 0)
{
    Running = value;
    if ((c_value == 'T') || (c_value == 'F'))
        ret_val = TRIGRUNNING;
}
else if (strcmp (variable, "Toofast") == 0)
{
    Toofast = value;
    if ((c_value == 'T') || (c_value == 'F'))
        ret_val = TRIGTOOFAST;
}
else if (strcmp (variable, "Brake") == 0)
{
    Brake = value;
    if ((c_value == 'T') || (c_value == 'F'))
        ret_val = TRIGBRAKE;
}
else if (strcmp (variable, "Activate") == 0)
{
    Activate = value;
    if ((c_value == 'T') || (c_value == 'F'))
        ret_val = TRIGACTIVATE;
}
else if (strcmp (variable, "Deactivate") == 0)
{
    Deactivate = value;
    if ((c_value == 'T') || (c_value == 'F'))
        ret_val = TRIGDEACTIVATE;
}
else if (strcmp (variable, "Resume") == 0)
{
    Resume = value;
    if ((c_value == 'T') || (c_value == 'F'))
        ret_val = TRIGRESUME;
}
else
    fprintf (stderr, "Could not read the input, must be a variable name.\n");

return (ret_val);
/* 
 * main program function 
 */

main (argc, argv)
int argc;
char *argv[];
{
    int result = CONDITION;
    int trig_event;

    CurState = OFF;

    ParseArgs (argc, argv);

    while (TRUE)
    { /* Run until EXIT is entered */

        /* Check the current state */
        if (CurStPersist == TRUE)
        {
            printf ("Current state = ");
            PrintState (CurState);
            printf ("\n");
        } else
        {
            if (FileIn == FALSE)
            {
                printf ("Enter current state, ");
                CurState = GetState ();
            }
        }

        result = GetNextInput ();
        while (result == CONDITION) /* Set condition variables until Exit */
        { result = GetNextInput (); /* or a trigger event. */

            if (result == EXIT)
                return 0;

            /* TRIGGER event was entered. */
            /* This case statement encodes the transition table. */
            trig_event = result;
            switch (CurState)
            {
            case OFF:
                switch (trig_event)
                {
                    case TRIGIGNITED:
                        if (Ignited == TRUE)
                        {
                            CurState = INACTIVE;
                            printf ("State change: OFF -> INACTIVE\n");
                        }
break;
}

if (CurState == OFF)
    printf ("No state change: OFF\n");
break;

/*****************************************************/
case INACTIVE:
    switch (trig_event)
    {
    case TRIGIGNITED:
      if (Ignited == FALSE)
      {
          CurState = OFF;
          printf ( "State change: INACTIVE --> OFF\n" );
      } 
      break;
    case TRIGACTIVATE:
      if (Activate == TRUE && Running == TRUE && Ignited == TRUE && Brake == FALSE)
      {
          CurState = CRUISE;
          printf ("State change: INACTIVE --> CRUISE\n");
      } 
      break;
    }
    if (CurState == INACTIVE)
        printf ("No state change: INACTIVE\n");
    break;

/*****************************************************/
case CRUISE:
    switch (trig_event)
    {
    case TRIGIGNITED: /* C4 */
      if (Ignited == FALSE)
      {
          CurState = OFF;
          printf ( "State change: CRUISE --> OFF\n" );
      } 
      break;
    case TRIGRUNNING: /* C5 */
      if (Running == FALSE && Ignited == TRUE )
      {
          CurState = INACTIVE;
          printf ("State change: CRUISE --> INACTIVE\n");
      } 
      break;
    case TRIGTOOFAST: /* C6 */
      if (TooFast == TRUE && Ignited == TRUE )
      {

CurState = INACTIVE;
printf ("State change: CRUISE --> INACTIVE\n");
}
break;
case TRIGBREAK: /* C */
    if (!Brake && Ignited && Running && TooFast)
    {
        CurState = OVERRIDE;
        printf ("State change: CRUISE --> OVERRIDE\n");
    }
    break;
case TRIGDEACTIVATE:
    if (!Deactivate && Ignited && Running && TooFast)
    {
        CurState = OVERRIDE;
        printf ("State change: CRUISE --> OVERRIDE\n");
    }
    break;
}
if (CurState == CRUISE)
    printf ("No state change: CRUISE\n");
break;

/*-----------------------------------------------*/
case OVERRIDE:
    switch (trig_event)
    {
    case TRIGIGNITED:
        if (!Ignited)
        {
            CurState = OFF;
            printf ("State change: OVERRIDE --> OFF\n");
        }
        break;
    case TRIGRUNNING:
        if (!Running && Ignited)
        {
            CurState = INACTIVE;
            printf ("State change: OVERRIDE --> INACTIVE\n");
        }
        break;
    case TRIGACTIVATE:
        if (Activate && Ignited && Running && Brake)
        {
            CurState = CRUISE;
            printf ("State change: OVERRIDE --> CRUISE\n");
        }
        break;
    }
case TRIGRESUME:
    if (Resume == TRUE && Ignited == TRUE &&
        Running == TRUE && Brake == FALSE)
    {
        CurState = CRUISE;
        printf ("State change: OVERRIDE --> CRUISE\n");
    }
    break;
}
if (CurState == OVERRIDE)
    printf ("No state change: OVERRIDE\n");
break;

} /* End switch (CurState)*/
} /* End while */
} /* End main */
6.8.1 Full predicate coverage level test case requirements: ........................................ 69
6.8.2 Transition Pair Coverage Level Requirements: ............................................... 77
6.9 Active Lateral Navigation Submode Test Cases .......................................................... 79
   6.9.1 Full predicate coverage level test case requirements: ........................................ 80
   6.9.2 Transition Pair Coverage Level Requirements: ............................................... 81
6.10 Active Lateral Approach Submode Test Cases ....................................................... 81
   6.10.1 Full predicate coverage level test case requirements: ...................................... 81
   6.10.2 Transition Pair Coverage Level Requirements: .............................................. 82
6.11 Active Vertical Mode Test Cases ............................................................................ 83
6.12 Flight Level Change Submode Test Cases .............................................................. 184
   6.12.1 Full predicate coverage level test case requirements: ...................................... 185
   6.12.2 Transition Pair Coverage Level Requirements: .............................................. 188
6.13 Altitude Select Mode Test Cases ........................................................................... 189
   6.13.1 Full predicate coverage level test case requirements: ...................................... 189
   6.13.2 Transition Pair Coverage Level Requirements: .............................................. 190
6.14 Altitude Select ENABLED Submode Test Cases ....................................................... 191
   6.14.1 Full predicate coverage level test case requirements: ...................................... 191
   6.14.2 Transition Pair Coverage Level Requirements: .............................................. 193
6.15 Altitude Select ACTIVE Submode Test Cases .......................................................... 193
   6.15.1 Full predicate coverage level test case requirements: ...................................... 194
   6.15.2 Transition Pair Coverage Level Requirements: .............................................. 195
6.16 Vertical Approach Mode Test Cases ...................................................................... 195
   6.16.1 Full predicate coverage level test case requirements: ...................................... 195
   6.16.2 Transition Pair Coverage Level Requirements: .............................................. 197
6.17 Vertical Approach ENABLED Submode Test Cases ................................................ 197
   6.17.1 Full predicate coverage level test case requirements: ...................................... 198
   6.17.2 Transition Pair Coverage Level Requirements: .............................................. 198

7 CONCLUSIONS ........................................................................................................... 199

8 ACKNOWLEDGMENTS .............................................................................................. 200

A APPENDIX: CRUISE CONTROL IMPLEMENTATION .................................................. 204