Overview: The LRM currently specifies different statistics for “cover property” depending on whether the argument is a sequence_expr or a property_spec, however it is not clear how to distinguish a property_spec from a sequence_expr.

This proposal introduces a new syntax “cover sequence” that should be used when the all-match statistics are desired. While this is not backwards compatible, it is believed that current users, if any, will be able to migrate easily, and that this will provide a much better long term solution.

The second thing that this proposes is to eliminate the requirement for failure counters on coverage of a property, since coverage is the intent and not bug detection. Implementations can still choose to provide it if desired without violating the standard.

CHANGE 16.14 Concurrent assertions

16.14 Concurrent assertions

A property on its own is never evaluated for checking an expression. It must be used within a verification statement for this to occur. A verification statement states the verification function to be performed on the property. The statement can be one of the following:

— **assert** to specify the property as a checker to ensure that the property holds for the design
— **assume** to specify the property as an assumption for the environment
— **cover** to monitor the property or sequence evaluation for coverage

Change Syntax 16-16 of 16.14

```
concurrent_assertion_statement ::=
        assert_property_statement
        | assume_property_statement
        | cover_property_statement
        | cover_sequence_statement
  .
  .
Cover_sequence_statement::=
        cover sequence ( [clocking_event ] [ disable iff ( expression_or_dist ) ]
                sequence_expr ) statement_or_null
```

Change A.6.10

```
concurrent_assertion_statement ::=
        assert_property_statement
        | assume_property_statement
        | cover_property_statement
        | cover_sequence_statement
  .
  .
Cover_sequence_statement::=
        cover sequence (
                [clocking_event ] [ disable iff ( expression_or_dist ) ] sequence_expr ) statement_or_null
```

Change 16.14.3 Cover statement

There exist two categories of cover statements, **`cover sequence`** and **`cover property`**. Both ~~To~~ monitor ~~sequences and other~~ behavioral aspects of the design for coverage. ~~the same syntax is used with the **cover** statement~~. The tools can gather coverage information about the evaluation and report the results at the end of simulation or on demand via an assertion API (refer to Clause 38). The difference between the two categories is that for sequence coverage(e.g. **cover sequence**), all-match semantics is required, whereas for property coverage (e.g. **cover property**), there is at most one match per attempt. When ~~the property for~~ the **cover** statement is successful, ~~the~~ an optional pass statement can be executed. The pass statement is executed in the Reactive region. ~~specify a coverage function, such as monitoring all paths for a sequence~~. The pass statement shall not include any concurrent **assert**, **assume**, or **cover** statement.

~~Coverage results are divided into two categories: coverage for properties and coverage for sequences.~~

For property ~~sequence~~ coverage, the statement appears as follows:

cover property (~~`sequence_expr`~~ `property_spec`) `statement_or_null`

The results of this coverage statement for a property shall contain the following:

— Number of times attempted
— Number of times succeeded (maximum of one per attempt)
~~— Number of times failed~~
— Number of times succeeded because of vacuity

In addition, `statement_or_null` is executed every time a property is being evaluated to true.
The coverage counters for success~~, failure~~ or vacuous success do not include disabled evaluations. The attempt counter includes the attempts which result in disabled evaluation.

For sequence coverage, the statement appears as follows:

cover sequence (
[clocking_event] [**disable iff (** expression_or_dist **)**] sequence_expr **)** statement_or_null

Results of coverage for a sequence shall include the following:

— Number of times attempted
— Number of times matched (each attempt can generate multiple matches)

In addition, `statement_or_null` gets executed for every match. If there are multiple matches at the same time, the statement gets executed multiple times, one for each match.

Note that the argument to **`cover property`** may be a sequence. Whether **`cover property`** or **`cover sequence`** is used is dependent on whether one match or multiple match semantics is desired.