1450.4 meeting minutes – 08/03/11

Attendees: Ernie Wahl, Markus Seuring, Jim O'Reilly
Not present: Ajay Khoche, Oleg Erlich, Paul Reuter

Agenda:

- IEEE Meeting Preamble (No discussion of proprietary information).
- Continue with discussion of section 7.6, Variables and Expressions. We'll review some of the issues discussed in previous meeting for which we had no quorum. Of primary focus is the proposed modifications to the Limits data type, and the (currently-named) Contain and NotContain operators of the Limits data type (see minutes from 7/27 meeting - in that meeting, it was suggested to change the Contain and NotContain operator to a more general Check operator, and to extend the comparison operators allowable to include an == and != operator as well.

Summary:

- With respect to line numbers in syntax doc dated Jul 20, 2011:
- Suggestion: add operators <, >, <=, >=, ==, != to literal Limits
  - CONSENSUS: maintain form (LoLim op ResultPlaceHolder op HiLim), to be able to easily spot differences between limits due to tabular representation.
  - CONSENSUS: use I and O for inside and outside limits

    Limits lims1 = (0.4V <= I <= 2.4V); // Inside limits
    Limits lims2 = (0.4V <= O <= 2.4V); // Outside limits
- CONSENSUS: Line 2068, 2139: use operator Check instead of Contain, DontContain
  - Case A: pass is inside limits

```
Boolean: (0.4V <= result && result <= 2.4V)   Limits: (0.4V <= I <= 2.4V);
Boolean: (0.4V <  result && result <  2.4V)   Limits: (0.4V <  I <  2.4V);
Boolean: (1.4V <  result && result <  1.4V)   Limits: (1.4V <  I <  1.4V); // Can't pass
Boolean: (result == 1.4V)                     Limits: (1.4V <= I <= 1.4V); // Impractical
```

  - Case B: pass is outside limits

```
Boolean: !(0.4V <= result && result <= 2.4V)  Limits: (0.4V <= O <= 2.4V);
Boolean: !(0.4V <  result && result <  2.4V)  Limits: (0.4V <  O <  2.4V);
Boolean: !(1.4V <  result && result <  1.4V)  Limits: (1.4V <  O <  1.4V); // Can't pass
Boolean: (result != 1.4V)                     Limits: (1.4V <= O <= 1.4V); // Impractical
```

- Line 980: Enum FailMode: use NONE instead of PASS ?
  - CONSENSUS: use PASS
- Re-evaluate:
  - Line 2114: do we need need crossover limits, i.e., lolim > hilim ? Ref: functional test: VOL > VOH OR enforce lolim <= hilim ?
  - CONSENSUS: enforce LoLim <= HiLim; if crossover limits are required in the future, add result placeholder C
- The above solutions address two issues:
  - Only one Limits operator is needed for checking a value to be either inside or outside a limits window - the previous solution required two operators – Contain() and NotContain().
  - Further, the above syntax permits not only the typical modes of operation (checking whether a value is inside a limits window – either single ended or double ended), but also whether a value is OUTSIDE a limits window, or whether it is either "equal to" or "not equal to" a limit value. We believe this addresses the use case posed by Ajay Khoche in previous meetings, and is biased towards the most common use cases.

Actions:

- Jim to rework Verigy example from STIL paper so that the STIL translation is consistent with its original Verigy code.

Reference documents (If logged into your google account, can edit. If not, can only view.)

- http://spreadsheets.google.com/ccc?key=0AoKiPr1I9LY9dF95dkhSTVVqOU5GbWJyWFNhY0JPX0E&hl=en
- Namespace resolution examples document: http://docs.google.com/Doc?docid=0AYKiPr1I9LY9ZGY4dmNjNTNfMGZkOGJ2bmZy&hl=en
- Scratchpad spreadsheet: https://spreadsheets0.google.com/ccc?key=tQ93VDnAZ-Cl9RFKpPrPDzw&authkey=COzyro8K&hl=en&authkey=COzyro8K#gid=0
- Scratchpad "Word" doc: https://docs1.google.com/document/d/1zVu2M8nTJsrm0nFbBhiuM8-YRt4ErYqdy_uSa3x3_T4/edit?authkey=CLrgwrsG#

Next meeting: 08/10/11

For reference STIL .4 information can be found at the IEEE STIL website: http://grouper.ieee.org/groups/1450/ (select the P1450.4 link from the table) or use the direct link http://grouper.ieee.org/groups/1450/dot4/index.html