1450.4 meeting minutes - 09/04/08

Attendees: Jim O'Reilly, Ernie Wahl, Bruce Parnas, Ajay Khoche

Not present: Doug Sprague, Kevin Coggins

Agenda:

- Preamble:
  - Record Meeting (*2) (call not recorded, since we did not have a quorum)
  - IEEE Meeting Preamble (No discussion of proprietary information).
- Discuss semantics of Stop and SetBinStop statements in context of Tests and Flows, vs. in FlowNodes.
- Reference documents: stil_example_d24_090408.txt (posted on web in "Docs for WG Meetings" section; latest syntax document (D24 - posted on web).
- Next Meeting 09/11/08.

Summary:

- Semantics of Stop/SetBinStop vs Exit.
  - "**Stop;** ": stop executing FlowNode code but bubble up through Flows or Tests, executing PostActions and PassActions or FailActions for the Tests and Flows. Do NOT execute any more FlowNodes which contain the Tests and Flows.
    - If Stop or SetBinStop is executed in Test or Flow PostActions or PassActions/FailActions
      - Execute no more PostActions or PassActions/FailActions for that Test or Flow.
      - Return to FlowNode which invoked the Test or Flow
      - Do not execute FlowNode PostActions or ExitPort selection and actions, but return from the FlowNode to the Test or Flow which caused execution of that FlowNode.
      - Continue with this "bubble-up" return through the various levels of hierarchy of Tests, FlowNodes, and Flows until reaching the Test or Flow specified by the EntryPoint which initiated the action. Returning through the levels, Test or Flow PostActions and PassActions/FailActions ARE executed wherever encountered; however, within a Test or Flow, no additional FlowNodes are executed (including the ExitPort selection and ExitPort actions).
    - If Stop or SetBinStop is executed in FlowNode PostActions or ExitPort Actions:
      - Execute no more FlowNode PostActions or ExitPort Actions, but return from the FlowNode to the Test or Flow which caused execution of that FlowNode.
      - Continue with this "bubble-up" return through the various levels of hierarchy of Tests, FlowNodes, and Flows until reaching the Test or Flow specified by the EntryPoint which initiated the action. Returning through the levels, no additional FlowNodes are executed (including the ExitPort selection and ExitPort actions); however, Test or Flow PostActions and PassActions/FailActions ARE executed wherever encountered.
  - **Exit <***integer_expr***>;** : jump directly to initiating EntryPoint, executing no Test or Flow PostActions nor PassActions/FailActions, and executing no more FlowNodes and FlowNode ExitPort selection and Actions. The pass/fail result returned to the EntryPoint initiator (which would normally be the ExecResult of the Test or Flow specified by the EntryPoint), is specified by the *integer_expr* token of the "**Exit <***integer_expr***>;**" statement. This statement is generally intended to be used for exceptions, such as hardware failures, which require immediate termination of test execution.
  - **Return;** : only allowed in FlowNode ExitPort blocks. Returns control to the beginning of the PostActions block of the Test or Flow containing the FlowNode whose ExitPort block issued

the Return statement. The ExecResult of the containing Test or Flow can be set in either the FlowNode ExitPort actions, prior to the Return statement, or from the PostActions of the Test or Flow containing the FlowNode which contains the Return statement. Execution continues as normal, with all FlowNode PostActions and ExitPort actions being executed, and all Test and Flow PostActions and PassActions/FailActions being executed, until reaching the EntryPoint, either via an Exit statement, a Stop statement, or by returning up the chain of Tests/Flows/FlowNodes.

- In definition of default FlowNode, allow the token NullExec in the TestExec statement. This is to allow definition of default FlowNodes when the actual block to be executed by the TestExec statement is not yet known. When a TestExec statement is encountered where a FlowNode would normally be expected, the contents of the default FlowNode are used instead, with the EXEC_OBJECT_NAME in the TestExec being substituted in place of NullExec.
- Updated definition of default FlowNode so that ExecResult of containing Test or Flow is set to Fail ONLY if the object being executed by that FlowNode failed.

```
FlowNode {
    PreActions { }
    TestExec EXEC_OBJECT_NAME;
    // Set ExecResult of containing Test or Flow unconditionally
    PostActions { ExecResult = CurrentExec.ExecResult; }
    ExitPorts {
        Port FAIL CurrentExec.ExecResult == Fail {
            SetBin CurrentExec.FailBin;
            Stop;
       }
       Port PASS  CurrentExec.ExecResult == Pass { Next; }
    }   // end ExitPorts
}   // end FlowNode
```

to

```
FlowNode {
    PreActions { }
    TestExec NullExec;
    PostActions { }
    ExitPorts {
        Port FAIL CurrentExec.ExecResult == Fail {
            // Set ExecResult of containing Test or Flow
            // to Fail only if a failure occurred.  Otherwise,
            // ExecResult of containing Test or Flow remains
            // unchanged from previous state.
            ExecResult = Fail;  // or ExecResult = CurrentExec.ExecResult
            SetBin CurrentExec.FailBin;
            Stop;
       }
       Port PASS  CurrentExec.ExecResult == Pass { Next; }
    }   // end ExitPorts
}   // end FlowNode
```

- STIL.4 definitions of TestBase and default FlowNode should be treated as good starting points; their contents should be appropriate for most common cases, but those objects can be modified by the user as appropriate. Further discussion, writing code examples, and feedback during the balloting process will indicate whether or not the default definitions are in fact appropriate.

- In FlowNode ExitPort expressions, allow the following syntax in the RHS of a boolean expression which uses the == comparison (in addition to general boolean expressions)

  <var_name> == 1,5:7

  This is interpreted as:

  <var_name> == 1 | <var_name> == 5 | <var_name> == 6 | <var_name> == 7

For example:

```
FlowNode {
    PreActions { }
    TestExec NullExec;
    PostActions { }
    ExitPorts {
        Port FAIL CurrentExec.ExecResult == -1,1,5:7 {
            ExecResult = Fail;
            SetBin CurrentExec.FailBin;
            Stop;
       }
       Port PASS  CurrentExec.ExecResult == Pass { Next; }
    }   // end ExitPorts
}   // end FlowNode
```

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