Use Case #4: Simple Two Test Flow (or Program Segment)

Author: Dave Dowding

Informative statement

This is a use case whose primary purpose is to be a vehicle to demonstrate/exercise the basic P1450.4 extension syntax and the flow entity interaction with the TestMethod

1.0 Priority - Must

1.1 This use case constitues a less-than "minimum viable" capability.

2.0 Assumptions/Prerequisites

2.1 This flow shows assumes that all aspects of the program are loaded and tester resources are initialized. If there are other flows to direct the loading of memories/registers and the initialization sequences, these must be discussed in an other Use Case.

2.2 No tester-to-DUT connect/disconnect sequences are encompassed in this flow.

2.3 The TestMethods assume the role of composing the PatternExec from the Timing/PatternBurst information (test criteria/parameters) passed from the flow data. DCLevels/Sets information (test specific setup parameters) are not passed from the flow data, but is encapsulated from within the TestMethods. [Note: this is one method of Test reuse.]

2.4 If Datalogging were required with this use case, no flow entity mechanisms are contained, implied or discussed herein. This is not a issue to be driven with this version of this use case.

2.5 It is intended that this sequence is run from start to termination

2.6 This sequence is focused for single site testing

3.0 Pre-Conditions

3.1 This flow has no dependencies on program are other than initial test pattern has been loaded and timing values established and tester resources are initialized. Pre-test settings of device power supplies and digital signal reference supplies and connectiviity with the DUT will be specified and controlled using the IEEE 1450.2-2002 DC_Levels extension constructs. These are not contained or defined within the flow specified in this use case.

4.0 Tasks/Scenario

4.1 **First FlowNode:** A Basic Functional Test ("wiggle test") at relaxed levels and timing.

4.1.1 Flow Node ID (Unique Name/Number) and any other entry actions taken (none.)

4.1.2 Parameters are passed from this flow node to Basic Functional TestMethod

4.1.2.0.1 DCLevels, DCSets and DCSequence are all set within the TestMethod and are therefore manufacturer/device family specific. See 1450.2-2002 for DCLevels constructs mentioned.)

(See IEEE 1450-1999 STIL Std spec, pgs 13-15, Figure 7, clause 5.2 for the next 2 items.)

4.1.2.0.2 Timing Domain Name "basic" is passed to the TestMethod

4.1.2.0.3 PatternBurst Name "basic_burst" is passed to the TestMethod

4.1.3 The Basic Functional TestMethod is called and "flow control" awaits TestMethod completion and a result condition of "Pass" or "Fail" returned.

4.1.4 FlowNode Post Actions: (none)

4.1.5 Test result returned to "flow control" and Flow Node Exit is determined

4.1.5.1 Pass (flow control arc to next flow node)

4.1.5.2 Fail Basic Functional (flow control arc to Basic Functional or wiggle Fail Bin) and terminate the program.

4.2 **Second FlowNode:** A Static Input Leakage test is accomplished by running a functional setup pattern to establish the appropriate levels on the DUT pins, and the input leakage is measured on the input pins. This is a simple test that can be implemented in a serial, ganged fashion with per-pin or system parametric test resource. The implementation is determined by the test methodology of the manufacturer for this device family. All determinations and limits are set within the TestMethod.

4.2.1 Flow Node ID (Unique Name/Number) and any other entry actions taken (none.)

4.2.2 Parameters are passed from this flow node to Static Input Leakage TestMethod

4.2.2.0.1 DCLevels, DCSets and DCSequence are all set within the TestMethod and are therefore manufacturer/device family specific. See 1450.2-2002 for DCLevels constructs mentioned.)

(The next 2 items are similar to those shown in IEEE 1450-1999 STIL Std spec, pgs 13-15, Figure 7, but varied to achieve correct state of the DUT for testing input leakages.)

4.2.2.0.2 Timing Domain Name "InLeakTimingSetup" is passed to the TestMethod

4.2.2.0.3 PatternBurst Name "InLeakFunc" is passed to the TestMethod

4.2.3 The Static Input Leakage TestMethod is called and "flow control" awaits TestMethod completion and a result condition of "Pass" or "Fail" returned.

4.2.4 FlowNode Post Actions: (none)

4.2.5 Test result returned to "flow control" and Flow Node Exit is determined

4.2.5.1 Pass Exit Node action (flow control arc to Good Device Pass Bin)

4.2.5.2 Fail Exit Node action (flow control arc to Static Input Leakage Fail Bin) and terminate the program.

5.0 Results/Post-Conditions

5.1 Appropriate actions can be taken by the Tester-Handler/Prober interface software to bin/mark/map the device relative to the bin results.

5.2 Appropriate power down and disconnect functions are not shown in this flow but are available to the user to determine correct sequences of those events using IEEE 1450.2-2000 DCLevels extension invocation not shown in this use case

5.3 Appropriate Tester-Prober/Handler interface interactions occur at appropriate points in the Device/Lot process. This may include handling of lot material and reports. These may also be selections determined at program load time.

6.0 Alternatives

6.1 None of note.

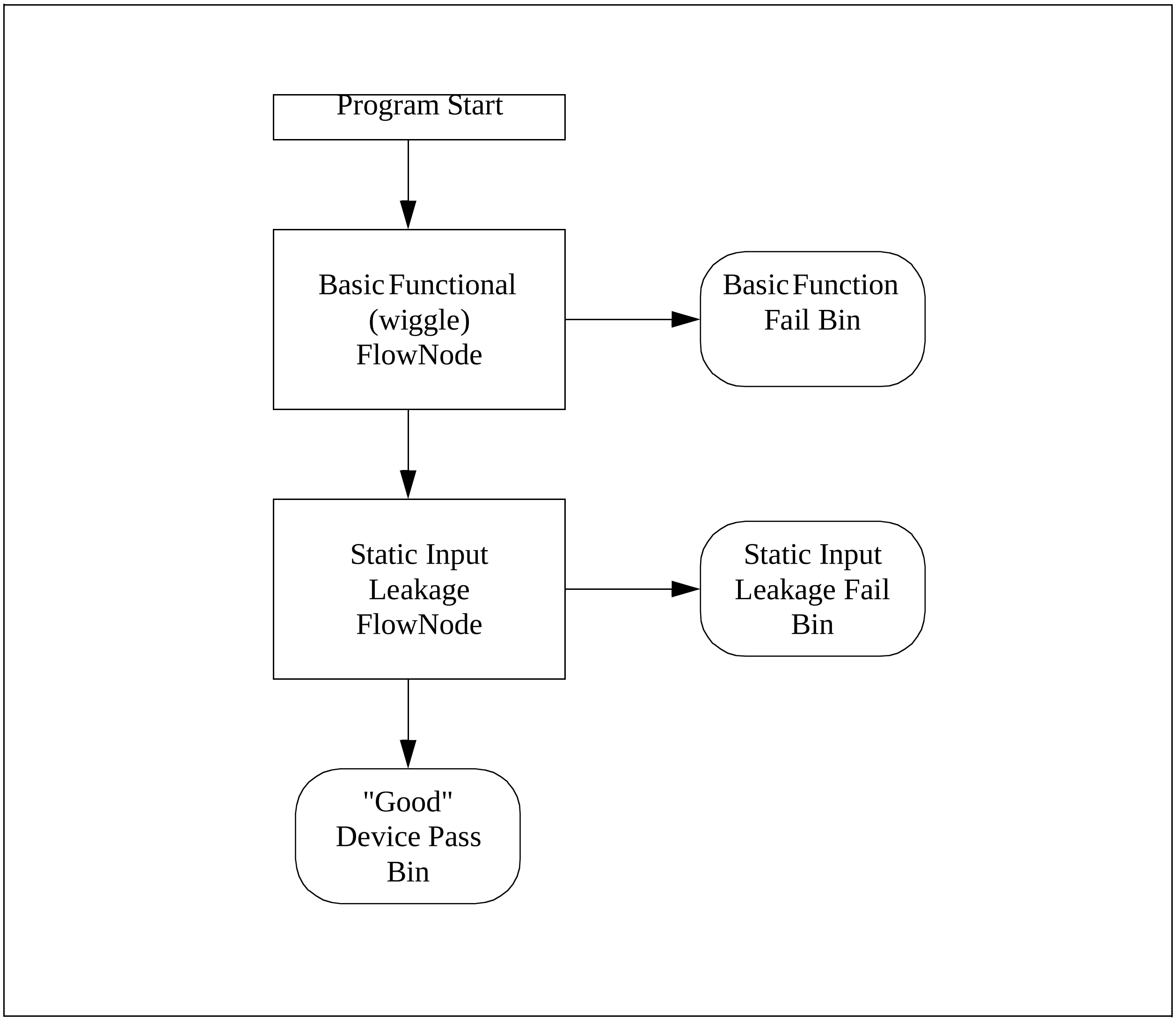
7.0 Comments/Questions

7.1 None.

8.0 P1450.4/STIL Syntax Example

8.1 None

9.0 Flow Diagram


Program Start
Basic Functional (wiggle) FlowNode
Basic Function Fail Bin
Static Input Leakage FlowNode
Static Input Leakage Fail Bin
"Good" Device Pass Bin

Note that the terminal nodes of this flow diagram are rectangles with rounded corners known in the industry as "Bins".