Use Case #1: Simple "Production" Program Flow

Document Revision: February 10, 2003 12:44 pm

Author: Dave Dowding

Informative statement

1.0 Priority - Must

1.1 Straight through flow to be supported by the P1450.4 extension

2.0 Assumptions/Prerequisites

2.1 This flow shows only the test program flow once all aspects of the program are loaded and tester requirements 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 This sequence is run from start to termination

2.4 This sequence is focused for single site testing

3.0 Pre-Conditions

3.1 This flow shows only the test program flow once all aspects of the program are loaded and tester requirements are initialized

4.0 Tasks/Scenario

4.1 **First FlowNode:** A functional continuity test is run, testing first for Shorts and then Opens. Upon functional continiuity failure, a parametric, per/pin test can be run. Whether this additional testing is dependent upon program setup. Subsequent parametric test fails may set runtime counters.

4.1.1 If this test passes, then the program flow passes to the Second FlowNode.

4.1.2 Flow Node information:

4.1.2.1 Flow Node ID (Unique Name/Number)Flow Node ID (Unique Name/Number)

4.1.2.2 TestMethod Parameters from flow to Company Standard Continuity TestMethod

4.1.2.3 Flow Node Exits

4.1.2.3.1 Pass (flow control arc to next flow node)

4.1.2.3.2 Fail Functional Continuity (flow control arc to Func_ContinuityFail Bin)

4.1.2.3.3 Fail DC Continuity (flow control arc to DC_ContinuityFail Bin)

4.1.3 Fail Flow Details

4.1.3.1 If the Function Continuity test fails:

4.1.3.1.1 Check program startup flag for "Skip_DC_Continuity.

4.1.3.1.2 If flag is set, then increment and check Consecutive Continuity Fail Counter. If the Consecutive Continuity Fail Counter limit is not exceeded, then bin the device as a Functional Continuity Fail and terminate the program execution.

4.1.3.1.3 If Consecutive Continuity Fail Counter limit is exceeded, halt the test program with a error message indicating this state and require operator intervention.

4.1.3.1.4 If flag is not set, then increment and check Consecutive Continuity Fail Counter. If the Consecutive Continuity Fail Counter limit is not exceeded, then start the DC_Continutity tests for Shorts and Opens on all pins in the continuity test list.

4.1.3.2 Run the DC Continuity for Shorts and Opens on all pins in the continuity test list.

4.1.3.2.1 If the DC Continuity test fails, bin the device as a Continuity fail, capture the pin/measured value info the measure for datalogging (if enabled) and then terminate the program.

4.1.3.2.2 If this DC Continuity Test passes, clear the Functional Continutiy fail flag and continue to the Second FlowNode

4.2 **Second FlowNode:** A Basic Functional Test (wiggle) at relaxed levels and timing. This may be a single Functional pattern or a PatternBurst list of patterns specific to verify the basic functionality and interconnectivity of the DUT/SoC.

4.2.1 Flow Node information:

4.2.1.1 Flow Node ID (Unique Name/Number)

4.2.1.2 TestMethod Parameters from flow to Basic Functional TestMethod (if any)

4.2.1.3 Flow Node Exits

4.2.1.3.1 Pass (flow control arc to next flow node)

4.2.1.3.2 Fail Wiggle Functional (flow control arc to WiggleFuncFail Bin) and terminate the program.

4.2.2 If this test passes, then program flow passes to the Third FlowNode.

4.2.3 If this test fails:

4.2.3.0.1 Bin the device as a Basic Functionality fail and terminate the program.

4.3 **Third FlowNode:** At Speed Functional Test, IP Cores 1, 3, 6 and 7. This test may be composed of a test method that runs the same patterns (PatternBurst) at minimum and then maximum VDD. Fail bins have been established for Fails at each IP Core and VDD_Min or for VDD_Max possible.

4.3.1 Flow Node information:

4.3.1.1 Flow Node ID (Unique Name/Number)

4.3.1.2 TestMethod Parameters from flow to Multi-Core Functional TestMethod (if any)

4.3.1.3 Flow Node Exits

4.3.1.3.1 Pass (flow control arc to next flow node)

4.3.1.3.2 Fail At Speed Functional IP_Core1-VDD_Min/Max (flow control arc specific Bin) and terminate the program.

4.3.1.3.3 Fail At Speed Functional IP_Core 3-VDD_Min/Max (flow control arc specific Bin) and terminate the program.

4.3.1.3.4 Fail At Speed Functional IP_Core 6-VDD_Min/Max (flow control arc specific Bin) and terminate the program.

4.3.1.3.5 Fail At Speed Functional IP_Core 7-VDD_Min/Max (flow control arc specific Bin) and terminate the program.

4.3.2 If this test passes, the program flow passes to the Fourth FlowNode.

4.3.3 If this test fails details:

4.3.3.1 Determine the VDD min or max state of the test and Bin the device as a appropriately for the IP Core At Speed functional fail and terminate the program.

4.4 **Fourth FlowNode:** At Speed Functional Test, IP Cores 2, 4, 5, 8, 9 and 10. This test may be composed of a test method that runs the same patterns (PatternBurst) at minimum and then maximum VDD (differing from those in the Third FlowNode). Fail bins have been established for Fails at each IP Core and VDD_Min or for VDD_Max possible.

4.4.1 Flow Node information:

4.4.1.1 Flow Node ID (Unique Name/Number)

4.4.1.2 TestMethod Parameters from flow to Multi-Core Functional TestMethod (if any)

4.4.1.3 Flow Node Exits

4.4.1.3.1 Pass (flow control arc to next flow node)

4.4.1.3.2 Fail At Speed Functional IP_Core2-VDD_Min/Max (flow control arc specific Bin) and terminate the program.

4.4.1.3.3 Fail At Speed Functional IP_Core 3-VDD_Min/Max (flow control arc specific Bin) and terminate the program.

4.4.1.3.4 Fail At Speed Functional IP_Core 5-VDD_Min/Max (flow control arc specific Bin) and terminate the program.

4.4.1.3.5 Fail At Speed Functional IP_Core 8-VDD_Min/Max (flow control arc specific Bin) and terminate the program.

4.4.1.3.6 Fail At Speed Functional IP_Core 9-VDD_Min/Max (flow control arc specific Bin) and terminate the program.

4.4.1.3.7 Fail At Speed Functional IP_Core 10-VDD_Min/Max (flow control arc specific Bin) and terminate the program.

4.4.2 If this test passes, then program flow passes to the Fifth FlowNode.

4.4.3 If this test fails Details:

4.4.3.1 Determine the VDD min or max state of the test and Bin the device as a appropriately for the IP Core At Speed functional fail and terminate the program.

4.5 **Fifth FlowNode:** Dynamic AC test(s) (i.e. Leakage, VOL/VOH, Output Leakage, etc.) The test is accomplished by running a functional setup pattern that stops at various points where pins are in the appropriate state to be tested. Then if the tests passes, the pattern is run progressively until all pins are tested. This flow node is not run at WaverSort.

4.5.1 If this run of the test program is at WaferSort, then go directly to the Sixth FlowNode.

4.5.2 If these tests pass, then program flow passes to the Sixth FlowNode.

4.5.3 If this test fails:

4.5.3.1 The device is Binned as a (AC_Test Name) fail, failing test information is capture (if datalogging enabled) and the program is terminated.

4.6 **Sixth FlowNode:** Operating Idd Test at VDD max. The device is setup to run at VDD max on a specific functional pattern and the Idd measurement(s) are taken and tested to specified limits.

4.6.1 If this test passes, then program flow bins the device as Good and program flow terminates.

4.6.2 If this test fails:

4.6.2.1 The device is Binned as a Idd-VDD_Max fail and the program is terminated.

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

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 The program can be loaded and setup to run as WaferSort, PackageTest, or QA_Test at load time.

7.0 Comments/Questions

7.1

8.0 P1450.4/STIL Syntax Example

8.1