Chip-level [Chipname] [Blockname] Test Plan Template

Author: [Name], Equator Technologies, Inc.

1.0 System Description - Required

1.1 Summary of System Functionality

Action: List and paraphrase the function of each new, changed or re-used block
Intent: Provide overview and scope of functions to be verified

Table 1: List of System Blocks including re-used, new, and changed

Block Name	Name of Function(s)	Status (New/ Changed)	Explanation of Function(s) to be tested
Block XY	Status Register A	Changed	3 new status bits; (1) last reset was a system reset, (2) transaction ongoing, and (3) transaction finished
	State machine B	New	Internal bus protocol state machine
Block Z	ABC Control Register bit 0	Changed	Reset value changed from 0 to 1

1.2 System Block Diagram

Action: Provide a block diagram of the Design Under Verification; use a color or shading system to identify re-used, changing, and/or new blocks.
Intent: Provide visual aid to "Summary of System Functionality" of Design Under Verification

1.3 Relevant DUV Specification Version

Action: Name the specification(s) and spec(s) version(s) used to create test plan
Intent: Provide a reference to the local or industry standard specification(s) used for verification-oriented interpretations

1.4 Previous Project(s) Verification Problem Area(s) Carry-forward

Action: List problem(s) encountered in verifying previous project(s)
Intent: Encourage resolution and/or warn of existing problem area

1.5 Acronyms Used in Test Plan

Action: Define acronyms used in test plan document
Intent: Provide clarity to the reader and documentation efficiency to the writer

2.0 Chip-level First-Pass Design Verification Success Criteria

Action: List first-pass verification success criteria in terms of testcase and testbench development

Intent: Plan for verification of design first pass success

2.1 Testcase Generation Plan

Action: Explain what new chip-level testcase generation will be required
Intent: Plan chip-level testcase need for the block under test

2.1.1 Current Testcases Update - Required

Action: Analyze current testcases for extent of re-usability
Intent: Provide prioritized list of testcases which will require update in order to be useful as verification software for the current design project

2.1.2 Testcase Types Plan for New Testcases - Required

Action: Plan for types of chip-level testing to be used in new testcases
Intent: Consider the design changes and new design and what is required to test it

2.1.2.1 Feature or Function

Action: Plan for the types of Feature/Functional testing to be employed or not and why
Intent: Arrive at an overview or vision of chip-level functional/feature testing needed for the block

2.1.2.1.1 Internal (check computation)

Action: Plan for Internal checking to be done or not done and why
Intent: Determine extent of need for Internal testing

2.1.2.1.2 External (check interface)

Action: Plan for External checking testcases to be done or not done and why
Intent: Determine extent of need for External testing

2.1.2.1.3 Debug Mode

Action: Plan for special debug modes testcases to be done or not done and why
Intent: Determine extent of need for debug mode testing

2.1.2.1.4 Characterization

Action: Plan for special characterization testcases to be done or not done and why
Intent: Determine extent of need for characterization testing needs

2.1.2.1.4.1 speed

Action: Plan for speed characterization testcases to be done or not done and why
Intent: Determine extent of need for speed characterization testing needs

2.1.2.1.4.2 timing

Action: Plan for timing characterization testcases to be done or not done and why
Intent: Determine extent of need for timing characterization testing needs

2.1.2.2 Performance

Action: Plan for special performance testcases to be done or not done and why
Intent: Determine extent of need for performance testing

2.1.2.3 Standards Conformance

Action: Plan for standards conformance testcases to be done or not done and why
Intent: Determine extent of need for standards conformance testing

2.1.2.4 Chip Initialization and special mode

Action: Plan for chip initialization testcases to be done or not done and why
Intent: Determine extent of need for chip initialization testing

2.1.2.5 Bug-related

Action: Plan for bug-related testcases to be done or not done and why
Intent: Determine extent of need for bug-related testing

2.1.2.6 CPU-host versus CPU-non-host mode testing

Action: Plan for testing in both host and non-host modes to be done or not done and why
Intent: Determine need for "Chip-master" vs. "Chip-slave" mode testing

2.1.2.7 Testcase-Centric Self-Checking - Required

Action: Plan for types of self-checking to be done or not done and why
Intent: Determine how self-checking will be done for the block

2.1.2.7.1 On-line (run-time result computation)

Action: Plan for appropriate on-line self-checking to be done or not done and why
Intent: Determine how on-line self-checking will be done for new diags, or added for diags which are not currently self-checking

2.1.2.7.2 Off-line (file-based, or pre-computed results)

Action: Plan for appropriate off-line self-checking to be done or not done and why
Intent: Determine how off-line self-checking will be done for new diags, or added for diags which are not currently self-checking

2.1.2.8 Changes or Additions to Testcase-centric PASS/FAIL Mechanism(s) - Required

Action: Explain the current method(s) for testcase-centric PASS/FAIL mechanism(s) and plan for changes or additions to be done or not done and why
Intent: Determine if any changes or additions need to be made to the current mechanism(s)

2.1.2.9 Testcase Programming Language

Action: Explain the choice of types of testcase coding to be employed and why
Intent: Indicate the type(s) of coding to be used with special attention to changes from the default, "tg", style of coding

2.1.2.9.1 Assembly (tg or tegger)

Action: Explain the choice of tg or tegger language or not and why
Intent: Indicate if tg or tegger style of coding is to be used and to what extent

2.1.2.9.2 C/C++

Action: Explain the choice of C/C++ language or not and why
Intent: Indicate if C/C++ language is to be used, why it is to be used, and to what extent

2.1.2.9.3 Verilog

Action: Explain the choice of verilog language for testcase generation or not and why
Intent: Indicate if verilog language is to be used, why it is to be used, and to what extent

2.1.2.9.4 Other Verification Coding Language

Action: Explain the choice of any other language not listed in this section and why
Intent: Indicate if any other language is to be used and to what extent

2.1.3 Explanation of Parts of Design Left Untested - Required

Action: List what is not to be tested and why
Intent: Justify or explain why parts of the design will not be tested

2.2 Chip-level Testbench Development Plan

Action: Plan for testbench development goals
Intent: Show how the testbench will change for the current project and to what extent

2.2.1 Testbench Goals

Action: List and explain development goals for the verification environment
Intent: Show a vision of what will be needed to perform block-oriented chip-level testing

2.2.1.1 New Testbench Development Goals - Required

Action: Show the new testbench development which will be needed to test the block
Intent: Define the boundaries between new and existing portions of the testbench

2.2.1.1.1 Testbench Harness[1] Development Goals

Action: Identify additions to overall testbench harness resulting from block changes and additions
Intent: Plan for testbench harness changes
Definition: the term harness refers that part of the testbench which once created changes the least if at all; this facility is where the component and interface signals are declared, where we instantiate the design under verification (DUV), and where interface signals are mapped to the ports of the design as well as to signals of the bus-functional procedures.

2.2.1.1.2 Testbench Utilities Development Goals

Action: Identify additions/changes to overall testbench utilities resulting from block changes, additions and methodology goals
Intent: Plan for testbench utilities changes

2.2.1.1.3 Monitors Development Goals (timing specs, bus monitors/checkers)

Action: Identify additions to overall testbench monitors resulting from block changes and additions
Intent: Plan for testbench monitors changes

2.2.1.1.4 Bus Functional Models Development Goals

Action: Identify additions to overall testbench bus functional models resulting from block changes and additions
Intent: Plan for testbench bus functional models changes

2.2.1.2 Existing Testbench Improvement Goals

Action: List improvement goals for the verification environment based on information from Section 1.4, “Previous Project(s) Verification Problem Area(s) Carry-forward,” on page 1
Intent: Acknowledge problems of the past and plan for incremental gains

2.2.1.2.1 Testbench Harness Improvement Goals

Action: Identify additions to overall testbench harness resulting from block changes and additions
Intent: Plan for testbench harness changes

2.2.1.2.2 Testbench Utilities Improvement Goals

Action: Identify additions to overall testbench utilities resulting from block changes and additions
Intent: Plan for testbench utilities changes

2.2.1.2.3 Monitors Improvement Goals (timing specs, bus monitors/checkers)

Action: Identify additions to overall testbench monitors resulting from block changes and additions
Intent: Plan for testbench harness changes

2.2.1.2.4 Bus Functional Models Improvement Goals

Action: Identify additions to overall testbench bus functional models resulting from block changes and additions
Intent: Plan for testbench bus functional models changes

---

1. **The term *harness* refers that part of the testbench which once created changes the least if at all; this facility is where the component and interface signals are declared, where we instantiate the design under verification (DUV), and where interface signals are mapped to the ports of the design as well as to signals of the bus-functional procedures.**

2.2.1.3 Testbench Programming Language

Action: Explain the choice of types of testcase coding to be employed and why
Intent: Indicate the type(s) of coding to be used with special attention to changes from the default, "tg", style of coding

2.2.1.3.1 PLI/C/C++

Action: Explain the choice of C/C++ language or not and why
Intent: Indicate if C/C++ language is to be used, why it is to be used, and to what extent

2.2.1.3.2 Verilog

Action: Explain the choice of verilog language for testcase generation or not and why
Intent: Indicate if verilog language is to be used, why it is to be used, and to what extent

2.2.1.3.3 Other Coding Language

Action: Explain the choice of any other language not listed in this section and why
Intent: Indicate if any other language is to be used and to what extent

2.2.1.4 Types of Testbench-centric Self-checking

Action: Plan for types of testbench-centric self-checking to be done or not done and why
Intent: Determine how the testbench will aid in self-checking for the block

2.2.1.4.1 Reference compare

Action: Plan for reference comparison self-checking to be done or not done and why
Intent: Determine how reference compare self-checking will aid in self-checking for the block

2.2.1.5 Testbench-centric PASS/FAIL Mechanism(s)

Action: Explain the current method(s) for testbench-centric PASS/FAIL mechanism(s) and plan for changes or additions to be done or not done and why
Intent: Determine if any changes or additions need to be made to the current mechanism(s)

2.2.1.6 Itemized list of expected output and tolerances

Action: Plan to address expected output and tolerances for those outputs as needed by the block
Intent: Test for expected outputs and tolerances thereof

3.0 Design Verification Deliverables

3.1 Previous Project(s) Applicable Coverage - Required

Action: Analyze current manufacturing, characterization, and functional coverage of each re-used or changing block
Intent: Determine the verification goal for each block and extent; and promote re-use

3.1.1 Functional Coverage Analysis

Action: Analyze current functional coverage
Intent: Provide a starting point for increasing functional coverage

3.1.1.1 Spec Annotation Function Matrix

Action: Analyze results of specification annotation
Intent: Provide thorough and prioritized list of testable functions in the specification

3.1.1.2 Current Testcase Coverage Matrix

Action: Link specification annotation result to current set of diagnostics.
Intent: Give account of functional coverage prior to the current project.

3.1.2 Manufacturing Production Coverage

Action: Analyze current functional coverage
Intent: Provide a starting point for increasing functional coverage

3.1.3 Silicon Characterization Coverage

Action: Analyze current functional coverage
Intent: Provide a starting point for increasing functional coverage

3.2 Current Project Testcase Coverage Matrix - Required

Action: Provide current project, or new, manufacturing, characterization, and functional coverage
Intent: Determine the verification goal for each block and extent; and promote re-use

3.2.1 Functional

Action: Link specification annotation result to current set of diagnostics.
Intent: Give account of current functional coverage.

3.2.2 Manufacturing Production

Action: Analyze current functional coverage
Intent: Provide a starting point for increasing functional coverage

3.2.3 Silicon Characterization

Action: Analyze current functional coverage
Intent: Provide a starting point for increasing functional coverage

3.3 Output logs - Required

Action: Provide location of testcase output logs
Intent: Offer data to substantiate PASS/FAIL claims

3.4 Ongoing Chip-level Summary Functional Coverage Report - Required

Action: Provide ongoing chip-level summary of overall functional coverage
Intent: Keep Equator team apprised of status and progress regarding functional coverage

3.5 Postmortem - Required

Action: Conduct a postmortem from the verification point-of-view
Intent: Conclude the verification project and learn from the experience

3.5.1 Project Validation/Verification Report

Action: Provide a document showing how the chip was tested and to what extent
Intent: Address the customer need and ISO9000 requirements for validation reporting per chip

3.5.2 List of Completed Deliverables

Action: Provide a list showing the deliverables that have been completed
Intent: Track status regarding completed deliverables

3.5.3 List of Uncompleted

Action: Provide a list showing the deliverables that have not been completed
Intent: Track status regarding uncompleted deliverables

3.5.4 List of Testbench Issues Resolved on this project

Action: Provide a list showing the issues that have been resolved from the last and current project
Intent: Track status regarding resolved issues

3.5.5 List of Testbench Issues Yet to be Resolved

Action: Provide a list showing the issues that have not been resolved from the last and current project
Intent: Track status regarding unresolved issues

3.5.6 Suggestions for Verification Methodology Improvements

Action: Provide a list of suggestions collected from team input for verification methodology improvement
Intent: Keep a record of those suggestions for verification improvement for the next project

4.0 Approvals - Required

4.1 Design Manager

4.2 Design Project Leader

4.3 Marketing

4.4 Applications/Driver Software Engineering