TWENTY-SEVENTH ANNUAL

TestConX™

Archive

DoubleTree by Hilton
Mesa, Arizona
March 1-4, 2026

A Full-Scenario Shmoo Solution for Enhancing Chip Test Efficiency

Lang Liu @AMLogic,
Chelsea Zhou & Steve Xie @Advantest
liulang25@mails.tsinghua.edu.cn, steve.xie@advantest.com,
chelsea.zhou@advantest.com

Mesa, Arizona • March 1–4, 2026

Agenda

- Background
- Scenarios of Shmoo Application
- Key Requirements & Capabilities
- Normal Shmoo Solution
- New Shmoo Solution
- Practical Example
- Summary

Background

- What is Shmoo?
  - An important tool in chip testing for visualizing performance boundaries
  - Maps chip behavior under varying input conditions (e.g., voltage, frequency)
- What kind of data can be obtained?
  - Directly parameters: Vmin/Vmax, maximum operating frequency
  - Indirect analysis data: Corner information, parallel pin test limits, hole distribution
- Which testing progress does it apply to?
  - Pre-production: Characteristic testing & Trim calibration
  - Mass production: Debug, correlation validation
  - Post-failure: Chip failure Analysis
  - Fast derive rough problem on hardware environment (like monitoring Vmin trend to catch test fixture degradation during production on socket or prober card)

Scenarios of Shmoo Application

- Covers pre-production to mass production, including calibration, debugging, and validation

Characteristic Test

Help engineers understand the chip's performance characteristics

Daily Debug

Help engineers quickly find and fix problems

Correlation Validation

To ensure the chip performs consistently under different conditions(even considering the bias between socket or prober card)

Trim Testing

To calibrate the chip's parameters for optimal performance

Key Requirements & Capabilities

Characteristic Test

- Requirement: Comprehensive method for data analysis
- Capability: Vmin/Vmax data collection, Shmoo hole detection, Frequency sweep , Standardized STDF output format

Daily Debug

- Requirement: Quick issue identification
- Capability: Flexible input/output format and multi-test item data collection

Correlation Validation

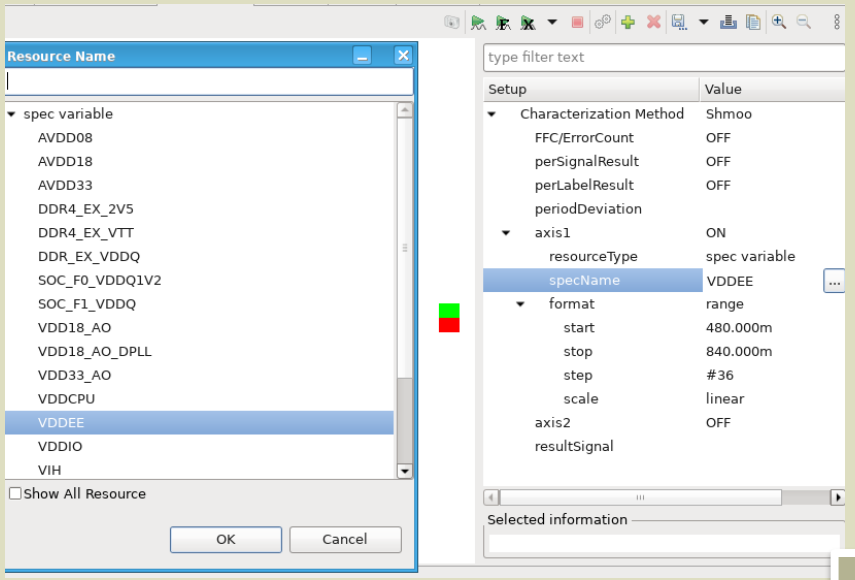
- Requirement: Collecting large amounts of data, deviation analysis
- Capability: Shmoo data overlay display & comparison function

Trim Testing

- Requirement: Trim data analysis
- Capability: Full-step ranking scan, best rank search

Original Shmoo Solution

- Solution Overview
  - Operating Foundation: SmarTEST 8 IDE native suite
  - Key Features: Quickly test spec search, fast step shmoo, and step-scale switching
- Example




```
shmoo shmooDDR_top_atspeed_comp_VDDEE_800M {
    target = DDR_top_atspeed_comp;
    resultTitle = "shmooDDR_top_atspeed_comp_VDDEE_800M";
    ffcErrorCount = false;

    axis[VDDEE] = {
        resourceType = specVariable;
        resourceName = "Spec_level.atspeed.levScan_Value.VDDEE";
        range.start = 0.48;
        range.stop = 0.84;
        range.steps = 36;
        range.fastSteps = 4;
        range.scale = linear;
    };
}
```

Shmoo Setup

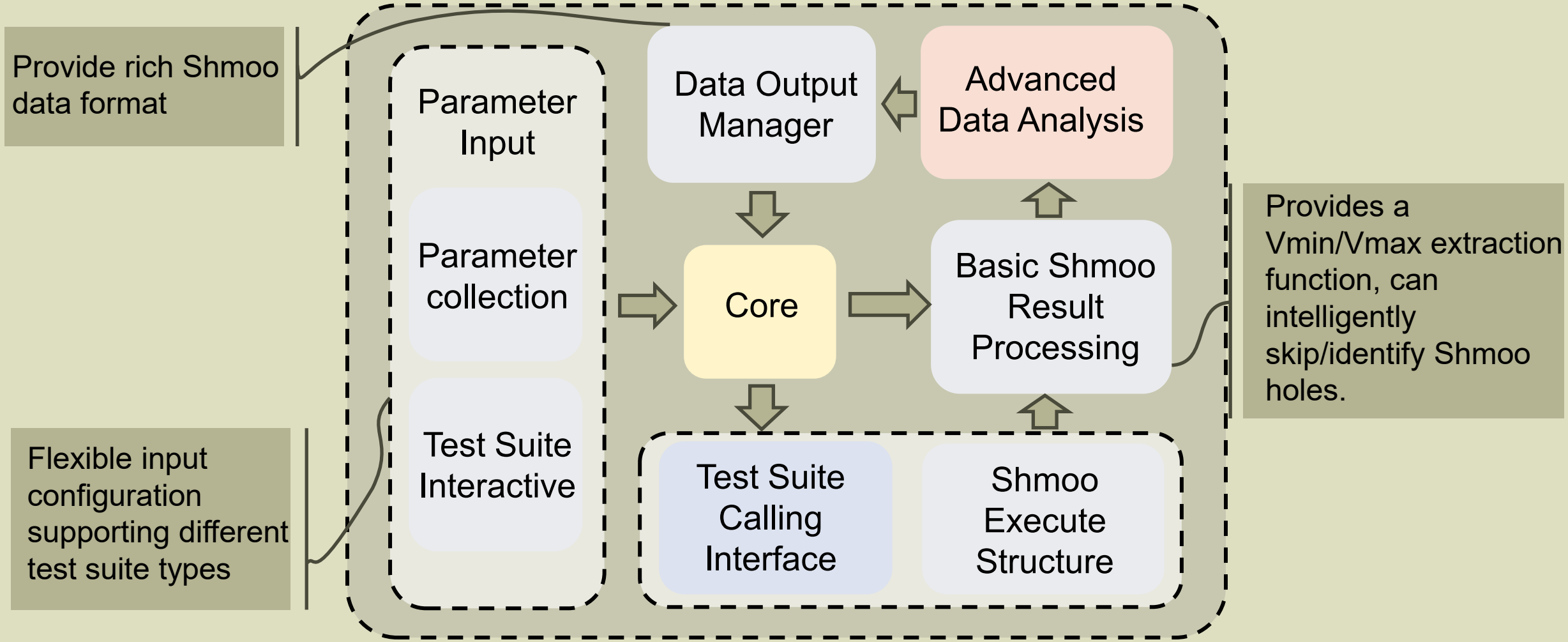
Original Shmoo Solution

- Input Constraints:
  - Limited to specification variables, no support for register or vector content.
- Limited Test Invocation:
  - Only support function test (e.g., Scan, Bist).
- Pass/Fail Evaluation:
  - Pass/Fail evaluation tied to whole test suite result (cannot checks test item result).
- No Deep Analysis:
  - Output data is difficult to understand and requires external tools (e.g., TP360, Vmin Report Tool)
- Output Constraints:
  - STDF logs provide only basic data, no advanced analysis (e.g., Shmoo hole detection).

New Shmoo Solution — Full-Scenario

- Solution Overview
  - Operation Foundation: SmarTEST 8 IDE enhanced interface
  - Key Features:
    - Scalable Structure
    - User-Friendly Operation
    - Flexible Input Configuration
    - Advanced Data Analysis
    - Clear and Effective Output
  - Capability:
    - Editable Data Output format
    - Multi-Test Item Data Collection
    - Shmoo Data Visualization & Comparison
    - Best Trim Setting Search

Full-Scenario Shmoo Solution



Full-Scenario Shmoo Solution

Overall Structure

```
PROCESS RunShmoo(input_params):
    Initialize configuration

    FOR EACH test_item IN suite:
        FOR EACH target_setting IN sweep_range:
            Apply target_setting
            Execute test_item
            Capture result (pass/fail, key measurements)
            Update summary metrics

    Perform Vmin/Vmax extraction with hole-skipping
    Generate standardized outputs (STDF, console summary, plots)
    RETURN {shmoo_matrix, Vmin, Vmax, summary_metrics}
```

Full-Scenario Shmoo Solution

Parameter Input

Flexible test item calling

Non-functional test

Functional test

+

Various parameter options

Voltage

Frequency

Register

Vector content

+

Optional judgment type

Function result

Global test result

Special test data

Flexible coverage of all test items with more comprehensive data collection.

Full-Scenario Shmoo Solution

Shmoo Execute Structure

```
double xStepValue = Math.abs(xAxis.start-xAxis.end)/(xAxis.step-1);
if (is2DMode) {
    yStepValue = Math.abs(yAxis.start-yAxis.end)/(yAxis.step-1);
}
int outerLoopCount = is2DMode ? yAxis.step : 1;
double xStartValue = (xAxis.start < xAxis.end) ? xAxis.start : xAxis.end;
double yStartValue = (yAxis.start < yAxis.end) ? yAxis.start : yAxis.end;
for (int y = 0; y < outerLoopCount; y += fastStep) {
    if (is2DMode) {
        yAxiValue = yStepValue*y+yStartValue;
    }
    result.set(new long[xAxis.step]);
    result_out.set(new String[xAxisList_new.size()]);
    temp = new String[xAxisList_new.size()];
    Arrays.fill(temp, " ");
    for (int x = 0; x < xAxis.step; x += fastStep) {
        double xAxiValue = xStepValue*x + xStartValue;
        result.setElement(x, new MultiSiteLong(0));
        if (is2DMode) {
```

1D&2D Shmoo Functionality Implementation

```
// fast step sweep
if (fs>1) {
    for (int si=0; si<activeSites.length; si++) {
        for (int j=fs; j<xSteps; j+=fs) {
            if (Math.abs(matrix.get(si,j) - matrix.get(si,j-fs)) > 0) {
                for (int x=j-fs+1; x<j; x++) {
                    double xv = xList.get(x);
                    logger.setCoordinates(x);
                    logger.startSuiteResults();
                    suite.setSpec(params.xAxis.specName, xv);
                    suite.execute();
                    MultiSiteBoolean pass = meas.isPass();
                    logger.endSuiteResults(); logger.send(pass);
                    matrix.set(si, x, pass);
                    if (onAfterPoint!=null) {
                        onAfterPoint.accept(x, 0);
                    }
```

Fast Step Mode Functionality Implementation

Full-Scenario Shmoo Solution

Basic Shmoo Result Processing

```
1        PROCESS ComputeShmooResults(matrix):
2            pass_band = contiguous_pass_band(matrix, skip_isolated_fails=True)
3            Vmin = pass_band.start
4            Vmax = pass_band.end
5            holes = detect_holes(matrix)  // after filtering isolated fails
6            RETURN {Vmin, Vmax, holes}
```

Accurate Vmin/Vmax extraction (hole skipping)

- **Accurate Vmin/Vmax extraction**
  Avoids incorrect result caused by isolated fail points.
- **Hole-Skipping Algorithm**
  Automatically detects and records 1-point “false fail” holes.
- **Cleaner Output**
  Generates standardized metrics for STDF, console view, and plotting.

Full-Scenario Shmoo Solution

Data Log

	Test#	Value	FFC	Unit	P/F
20MHZ_CORE0_shmoo.ptd_Vmin_first	160000605	0.710000			Pass
K_1920MHZ_CORE0_shmoo.ptd_Vmin	160000606	0.740000			Pass
K_1920MHZ_CORE0_shmoo.ptd_Vmax	160000607	1.06000			Pass
LK_1920MHZ_CORE0_shmoo.ptd_hole	160000608	1.00000			Fail

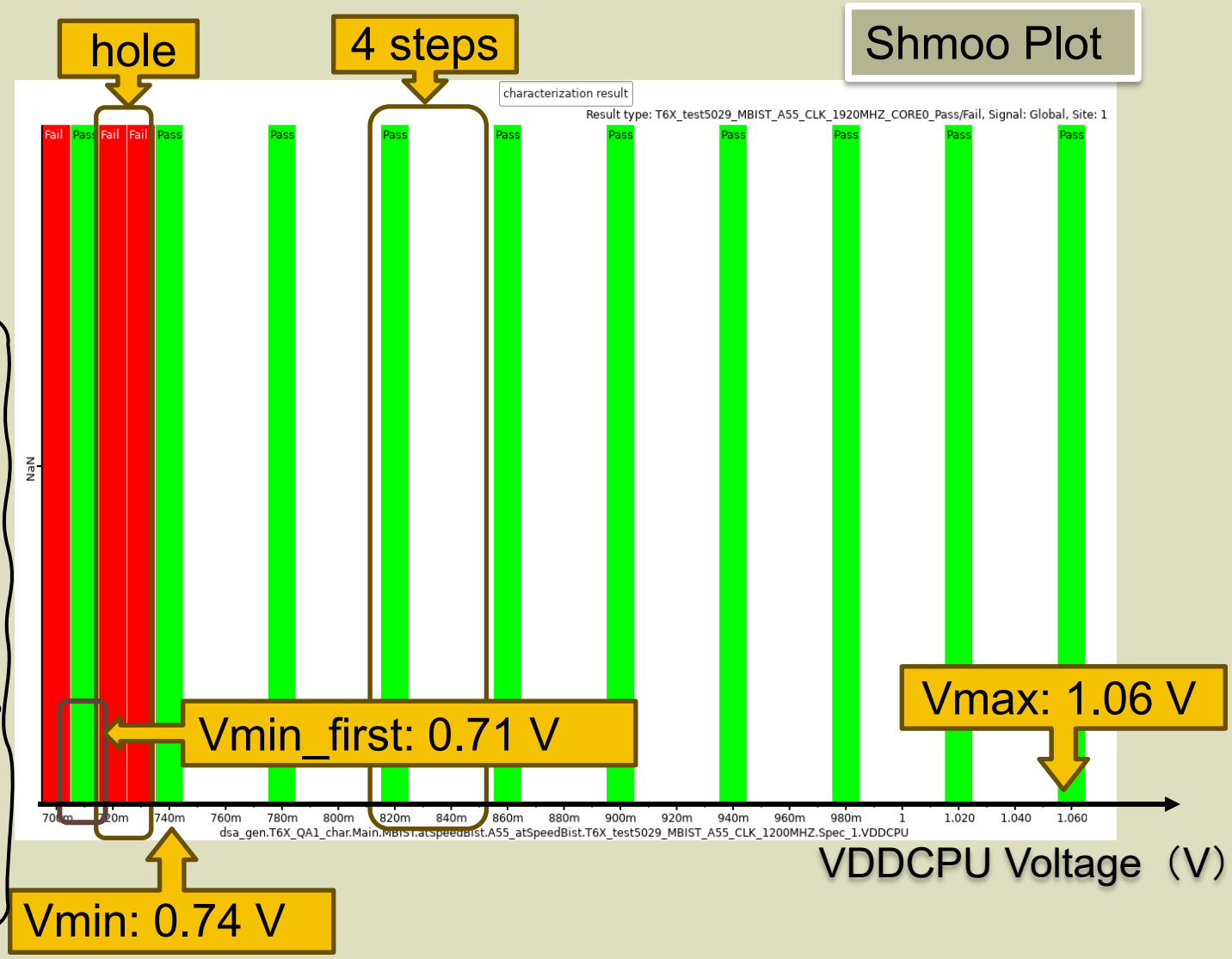
Fast step mode

- 4 step

Output data

- Vmin: Minimal value of continually pass region
- Vmax: Maximum value of continually pass region
- Hole: Fail vale between two pass region
- Vmin_first: Minimum pass value

Shmoo Plot



Full-Scenario Shmoo Solution

Data Output Manager

```
Target testsuit: IDD_Test
VDDEE Vmin_first: [4: 0.7]V
VDDEE Vmin: [4: 0.7]V
VDDEE Vmax: [4: 0.9]V
Site  TNum       Test            p/f       Signal  LoLim      Measure       HiLim
4     160000015  ptd_Vmin_first  Passed    N/A     N/A     <=   0.7      <=   N/A
4     160000012  ptd_Vmin        Passed    N/A     N/A     <=   0.7      <=   N/A
4     160000013  ptd_Vmax        Passed    N/A     N/A
4     160000014  ptd_hole        Passed    N/A       0
```

STDF Output

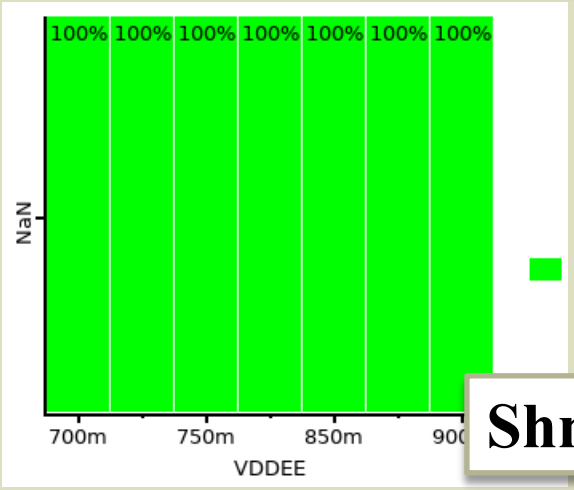
```
=======VDDCPU: 0.91V=======Periods: 4.1668E-8s=======
Global Result: Pass
Function Result: [1: true]
Frequency Result: [1: 953971.7419021933]
=======VDDCPU: 0.88V=======Periods: 4.1668E-8s=======
Global Result: Pass
Function Result: [1: true]
Frequency Result: [1: 953971.7419021933]
=======VDDCPU: 0.85V=======Periods: 4.1668E-8s=======
Global Result: Pass
Function Result: [1: true]
Frequency Result: [1: 953983
```

Console display

```
Target testsuit: IDD_Test
VDDEE Vmin: [1: 0.7, 2: 0.7, 3: 0.7, 4: 0.7, 5: 0.7, 6: 0.7, 7: 0.7, 8: 0.7]V
VDDEE Vmax: [1: 0.9, 2: 0.9, 3: 0.9, 4: 0.9, 5: 0.9, 6: 0.9, 7: 0.9, 8: 0.9]V

PreBindTest Suite Name = Main.IDD.IDD_Test:
Pattern = dsa_gen.T6X_QA1_char.Main.IDD.IDD_Test.OpSeq_2
DEVICE_NUMBER:1
Cur_Roff_SPEC:VDDEE = 0.800V
Roff_LL_ADJ:NaN
Roff_UL_ADJ:NaN
P        P        P        P        P        P        P
0.700    0.720    0.750    0.800    0.850    0.880    0.900    :VDDEE(V)
Vmin:0.700V
Vmax:0.900V
```

Summary text file

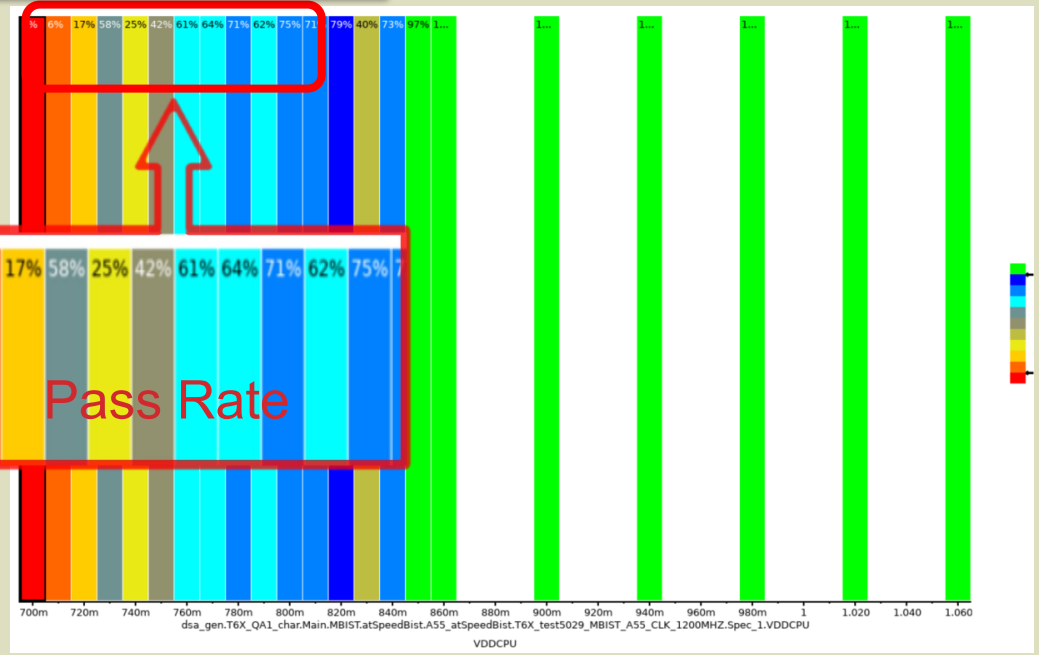


Shmoo plots

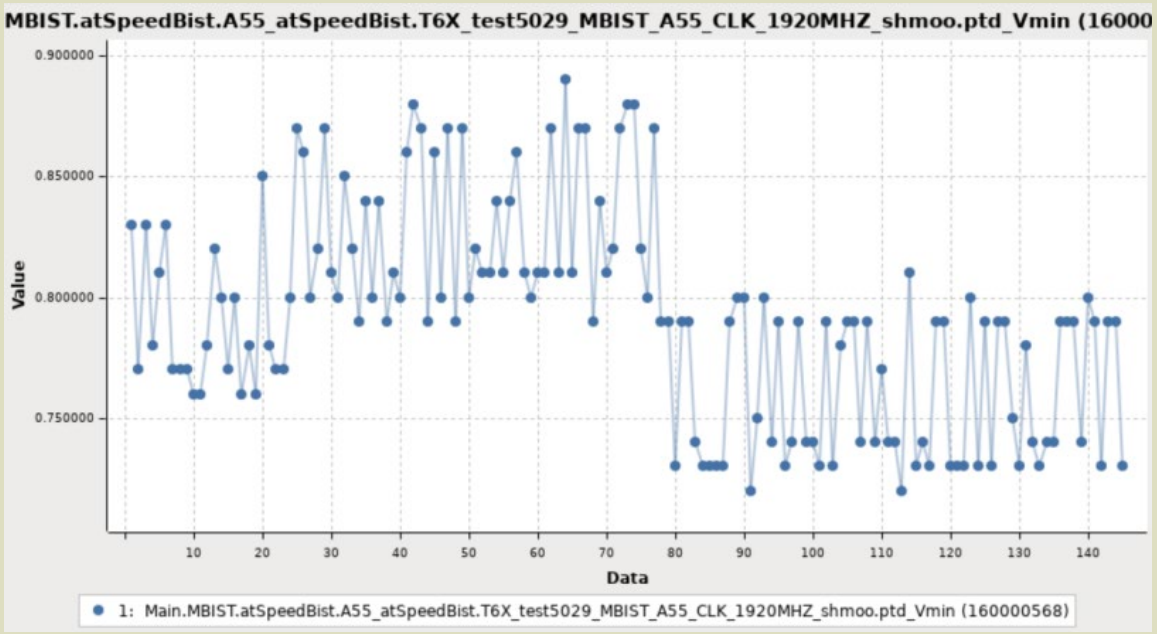
Full-Scenario Shmoo Solution

Advanced Data Analysis：Characteristic Test With Large Data

Shmoo Overlay



Vmin Trend Plot

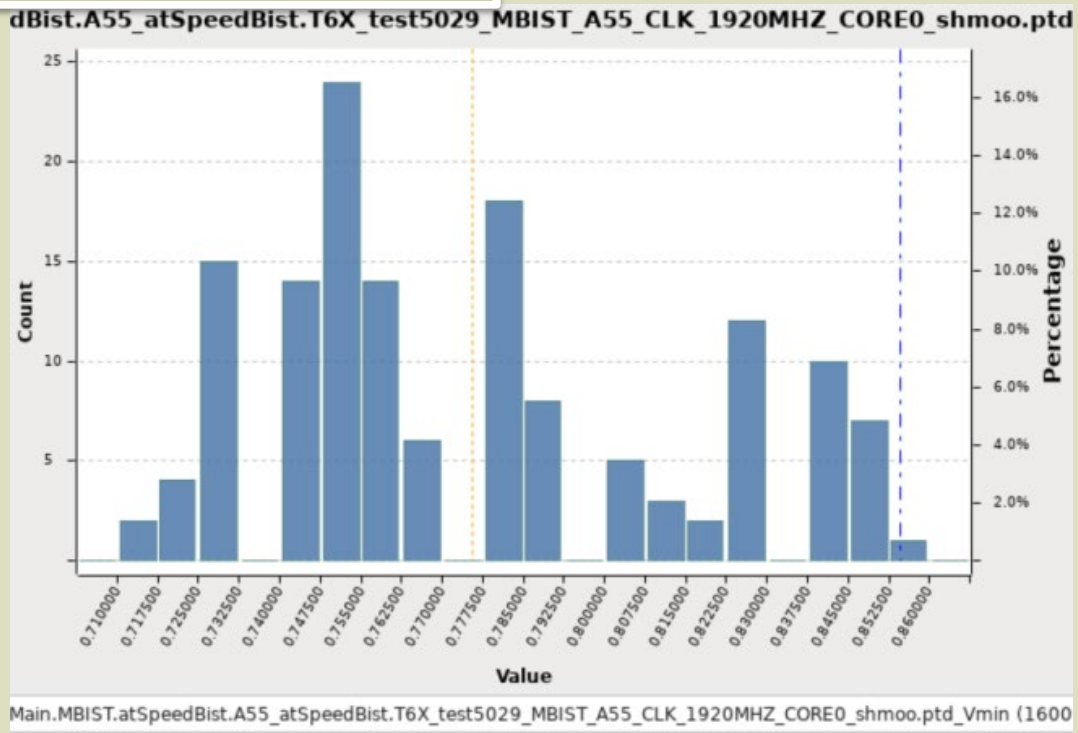


The data distribution is clearly displayed.

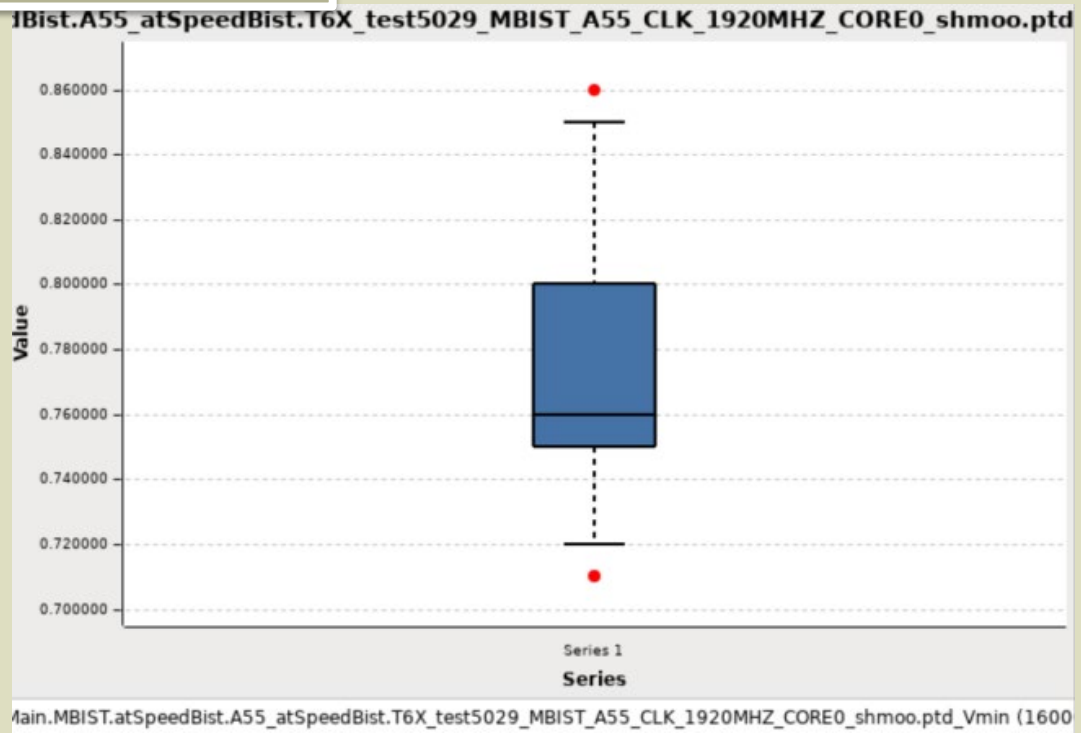
Full-Scenario Shmoo Solution

Advanced Data Analysis：Characteristic Test With Large Data

Vmin Histogram Plot



Vmin Box Plot



Practical Example

Input Parameter In Each Scenarios

Characteristic: 50+ Test Item Vmin/Vmax Test

Daily Debug: Change Register, output Frequency

Trim Test: Change Register/Vector Content, Judge Resistance Value

Correlation Validation: Change Vector Content, output Resistance Value

Scenario	Original Solution Setup Time	New Solution Setup Time	Improvement
Characteristic	8 hr	2 hr	-75%
Daily Debug	25 min/test	8 min/test	-68%
Trim Test	Manually modify method	Auto Full step scan	70% faster
Correlation Validation	Manually modify method	Auto modify parameter	60% faster

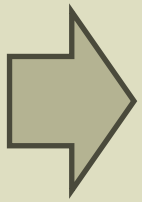
Practical Example

- BEFORE:
  - Engineer manually creates non-functional testing methods.
  - Manual assessment of the hole condition
  - Plots graphs by scripts
  - Review time ≈ 15 min/DUT
- AFTER:
  - Full-automatic specification modification
  - Modular structure makes adding or modifying code more efficient
  - Auto-hole-filter + summary result
  - Comprehensive summary plots
  - Review time ≈ 3 min/DUT

This reduced analysis time by 50%, saved ~15 engineering hours per lot.

Summary

- BEFORE:
  - Only function test supported
  - Hard-to-read STDF
  - Limited analysis capability



- AFTER (Full Scenario):
  - Multi-test type support
  - Clear data visualization
  - Automated Vmin/Vmax extraction

- Advantage:
  1. The type of test suite scan coverage from 50% up to 90%
  2. Cause of rich shmoo data format, data correlation time reduced by 50%
  3. With the highly customized execution framework, offline build efficiency improved by 70%

Acknowledgements

- Thanks to the ADVANTEST R&D team for technical support

- Gratitude to TestConX Technical Program Committee

COPYRIGHT NOTICE

www.testconx.org