Tutorial A:

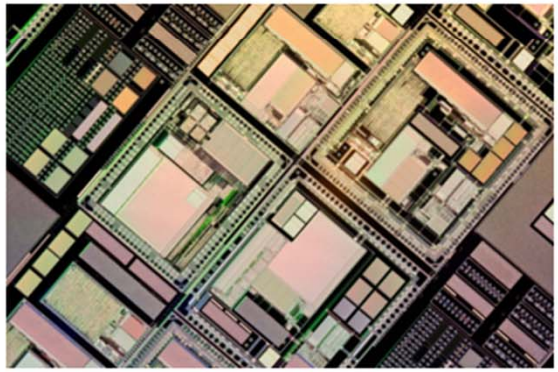
Analog Coverage – Yesterday‘s Dreams, Today‘s Reality and Tomorrow‘s Capabilities

Introduction

Walter Hartong, Lars Hedrich, Markus Olbrich

Analog/Mixed-Signal Verification: Challenges

- Increasing complexity…
  - More integrated functions (operation modes, frequencies, …)
  - More potential failure reasons (coupling, spurious tones, …)
  - Many different environmental conditions
  - Infinite parameter and property spaces



- …demands new approaches
  - More automation of verification
  - Coverage Measures

Why is the classical analog verification approach running out of steam?

- Simple verification/simulation plan might looks like:

Process Corners
nominal
ss
ff
fs
sf
…

×

Config	
schematic	extracted

×

Temperature			
-30	0	27	100

×

Param/Var/Conditions			
Vdd		Load	
1.1	1.3	Low	High

- However in reality …

© 2018 Cadence Design Systems, Inc. All rights reserved.

Growing verification challenges



Safety

Security



Analog Verification

Traceability

Quality





© 2018 Cadence Design Systems, Inc. All rights reserved.

Analog verification complexity
Flash Memory Interface Example

- 4 main blocks: receiver, transmitter, voltage reference, calibration
  - Each block contains ~50 electrical parameters
- Receiver modes / tests
  - CML or CMOS mode / single-ended and differential
  - Different input common-mode voltages, amplitudes, output loads
  - Distinct tests
    - Duty-cycle distortion, input to output delay, startup time, power-supply jitter curves, deterministic jitter, input referred offset, receiver threshold levels and hysteresis in CMOS mode, etc.
- Common performance and functional simulations
  - EM/IR, reliability asserts, aging and self-heating, power supply sequencing, electrical overstress
  - power down mode, ATB verification, digital interface timing, power and leakage, supply noise sensitivity, self-induced supply noise, supply inrush currents, etc.

© 2018 Cadence Design Systems, Inc. All rights reserved.

Analog verification complexity
Flash Memory Interface Example cont.

- Simulation based on pre-layout and extracted views
- All PVT corners needed
- Monte Carlo mismatch with global corners (TTG, SSG, FFG, SFG, and FSG)
- Deep sub-micron subjected to large variation and mismatch
  - requiring systematic calibration per-corner basis / per-MC-iteration
  - Impact of post-calibration voltage and temperature drift must be assessed
  - Combining thin and thick oxide devices, as well as core and I/O supplies greatly increase the number of corners
- Verification space is not trivial
  - reliability, functional, and performance sims require a distinct set of corners and operating conditions
  - power-up sequencing and other tests force fewer corners due to sim performance

© 2018 Cadence Design Systems, Inc. All rights reserved.

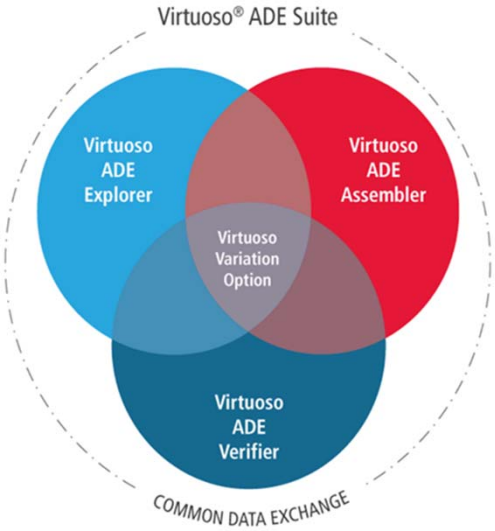
Subset of the Receiver verification/simulation plan

Sim Type	Simulation Name	Typical				Corners				Simulation Description	Corners
		Schematic		Post-Extract		Schematic		Post-Extract			
		Standard	Monte-Carlo	Standard	Monte-Carlo	Standard	Monte-Carlo	Standard	Monte-Carlo		
TRAN	Power supply sequencing	Done	N/A	To Do	N/A	Done	N/A	To Do	N/A	Simulate all combinations of power supply ramp-up/ramp-down sequences; check for excessive supply currents & electrical overstress (as for assert sims). Do a supply ramp-up spot check with 10us supply r/f times and check for overstress	Full PVT
TRAN	Functional	Done	N/A	Done	N/A	N/A	N/A	Done	N/A	Verification of all functional modes.	SS, FF, extreme VT
DC	ATB verification	Done	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Functional verification of ATB.	Typical
TRAN/DC	Power	Done	N/A	To Do	N/A	Done	N/A	To Do	N/A	Active power in all modes.	Power corners
TRAN/DC	leakage and Pad leakage	Done	N/A	Done	N/A	N/A	N/A	To Do	N/A	Leakage in disabled mode.	Full PVT
TRAN	EM/IR	N/A	N/A	N/A	N/A	N/A	N/A	To Do	N/A	EM and IR verification, in normal operations and power cycling.	EM/IR corner
RELCHECK	Reliability Asserts	Done	N/A	Done	N/A	Done	N/A	N/A	N/A	Check devices for electrical overstress in all operating modes.	EM/IR V&T; vary process
DC/TRAN	NBTI Asserts	Done	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Run the NBTI flow for all operating modes, including power-down modes.	T = 25C; Vary P & V
TRAN	Aging & Self-Heating	Done	N/A	N/A	N/A	To Do	N/A	To Do	N/A	Run a combined aging & self-heating at EM V&T and WC process.	EM/IR V&T; vary process
TRAN/AC	Supply noise sensitivity	Done	N/A	Done	N/A	Active	N/A	Active	N/A	Simulate the circuit with supply noise applied.	Full PVT
TRAN	Startup	Done	N/A	Done	N/A	To Do	N/A	To Do	N/A	Simulate re-enabling the circuit.	Full PVT
TRAN	DCD	Done	Done	Done	Done	Active	Active	Active	To Do	Using a 110011 bit pattern at 533Mbps, measuring output DCD. Both single-ended and differential modes.	Full PVT
TRAN	DCD monotonicity over temp	Done	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Using a 110011 bit pattern at 533Mbps, verify that the output DCD is monotonic under a full range temperature sweep.	TT
TRAN	DDJ	Done	N/A	N/A	N/A	Done	N/A	Done	N/A	Using a PRBS7 bit pattern at 533Mbps, measuring output jitter.	Full PVT
DC	input refered offset	Done	Done	N/A	Done	To Do	N/A	To Do	N/A	DC simulation to determine input refered offset, with and without calibration.	Typical and VT drift
AC	Input Capacitance	Done	N/A	Done	N/A	Done	N/A	Done	N/A	Input set at VCM, AC signal sent and AC current measuret to assess input capacitance.	PVT
TRAN	Calibration noise sensitivity	N/A	N/A	Done	N/A	N/A	N/A	Done	N/A	Perform offset calibraiton simulation in the presense of wideband noise.	TT snd PSIJ WC
TRAN	Uncalibrated operation	Done	N/A	N/A	N/A	Done	N/A	Done	Done	Run slow ramp on inp to assess VIH/VIL levels. Run same VIH/VIL MC on PVT worst cases to determine max total error and confirm it is lower than specified sensitivity. Run DCD MC sims on VIH/VIL worst case corners.	Full PVT
TRAN	Delay variation (lane to lane)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Done	Run Rx with input clock and measure delay. PVTR codes should remain the same even in MC sims as would be the case in a real system.	SS
TRAN	Offset-correction settling delay during calibration	N/A	N/A	N/A	N/A	Done	N/A	Done	N/A	Delay from calibration code change to Rx output toggle. Verify delay and impact (code difference) if not settled.	Full PVT
NOISE	Input referred noise	Done	N/A	N/A	N/A	Done	N/A	N/A	N/A	Input refered noise. First 2 stages only taken into account in order to prevent output foltage settling that would cause gain collapse and false high input noise reading.	Full PVT
TRAN	CMOS_DCD	N/A	N/A	Done	N/A	N/A	N/A	To Do	N/A	Using a 1010 bit pattern at 200Mbps, measuring the output DCD. Using input signal at VIL-VIH, not rail-to-rail.	Full PVT
TRAN	Hysteresis	Done	N/A	Done	N/A	Done	N/A	To Do	N/A	Measure the hysteresis voltage in the CMOS mode receiver (Vhst = Vih - Vil).	Full PVT
TRAN	Supply noise sensitivity	Done	N/A	N/A	N/A	Done	N/A	To Do	N/A	Simulate the circuit with supply noise applied. Using a 1010 bit pattern at 200Mbps, with noise at 133MHz in CMOS mode.	Full PVT
TRAN	DDJ_CMOS	Done	N/A	N/A	N/A	Done	N/A	To Do	N/A	Using a PRBS7 bit pattern at 533Mbps, measuring output jitter. In CMOS mode.	Full PVT

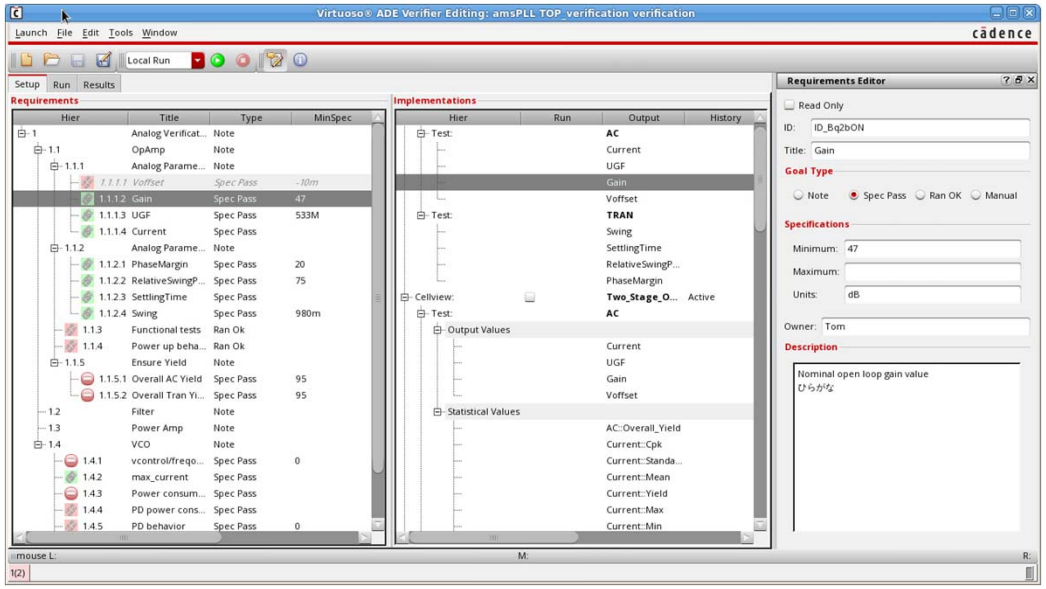
■ ■ ■

© 2018 Cadence Design Systems, Inc. All rights reserved.

Virtuoso ADE Verifier

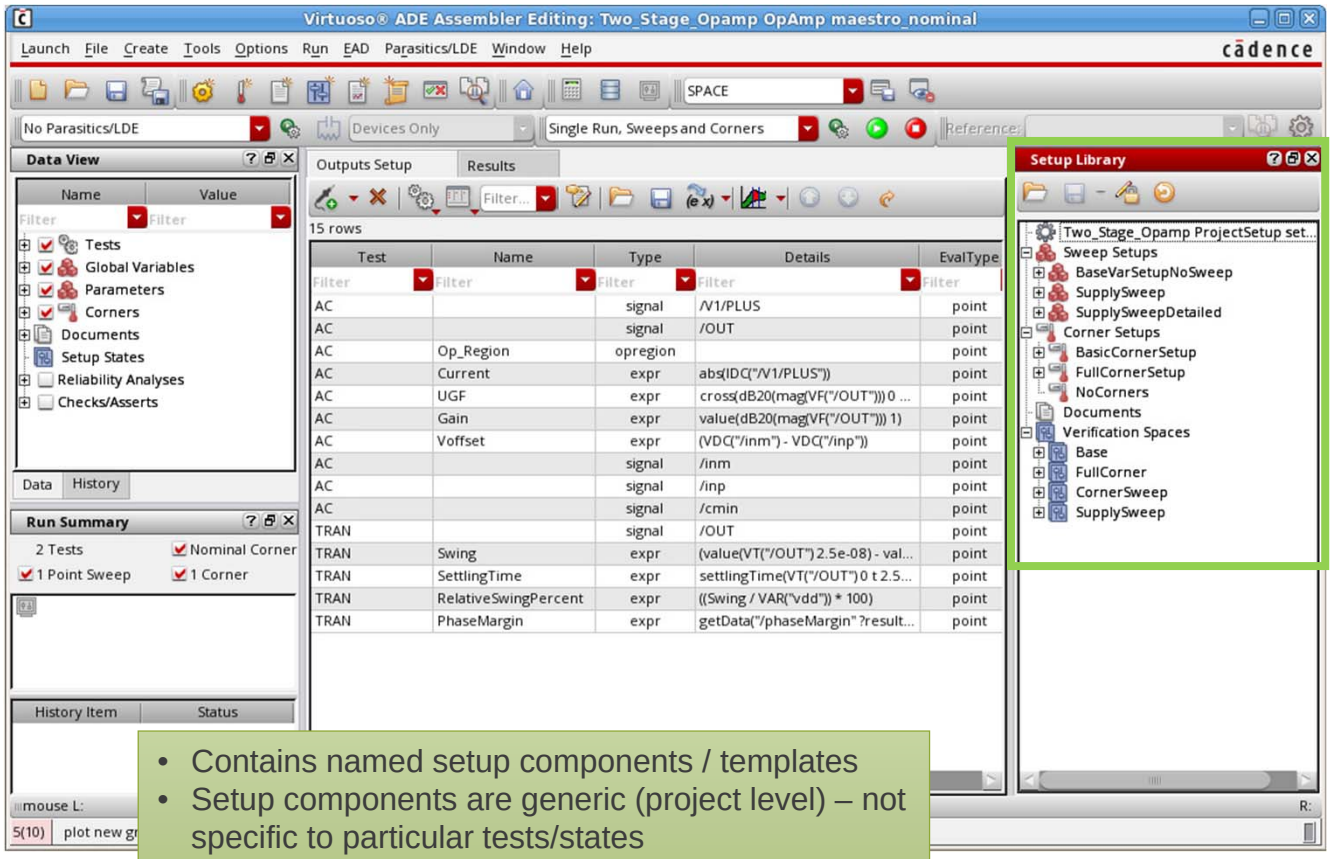


- A cockpit to drive plan based verification for analog designs
- Top down requirements driven analog verification flow
- Regression running capabilities enable more automated verification
- Requirements based reports/pass/fail/summary table to track progress
- Support customers needs for requirements tracking (ISO26262)
- Link analog verification to requirement management and digital verification tools



© 2018 Cadence Design Systems, Inc. All rights reserved.
cadence

Setup Library Assistant (SLA)

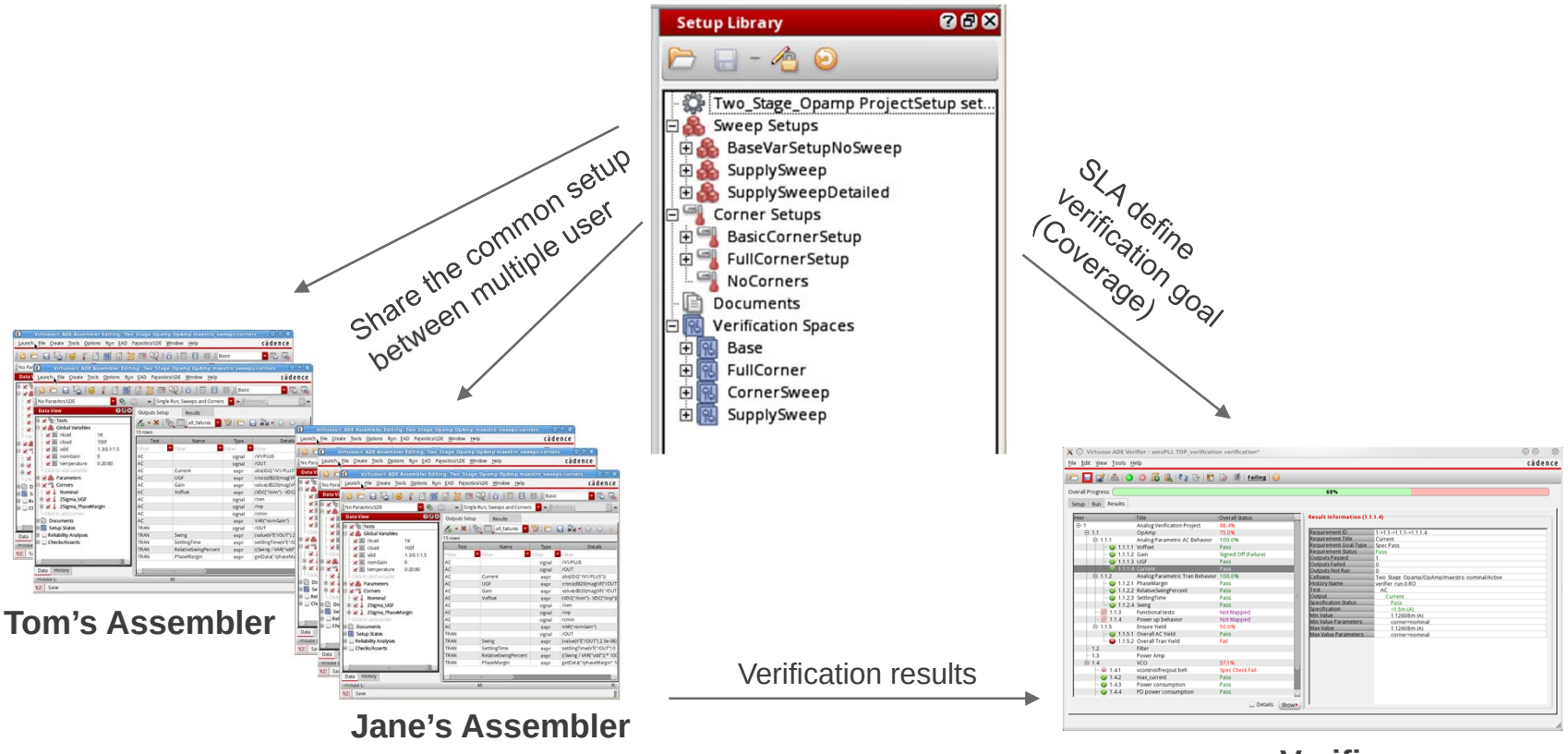


- Contains named setup components / templates
- Setup components are generic (project level) – not specific to particular tests/states
- SLA setup is save in a separate cellview

© 2018 Cadence Design Systems, Inc. All rights reserved.
cadence

Shared Setup Library Assistant

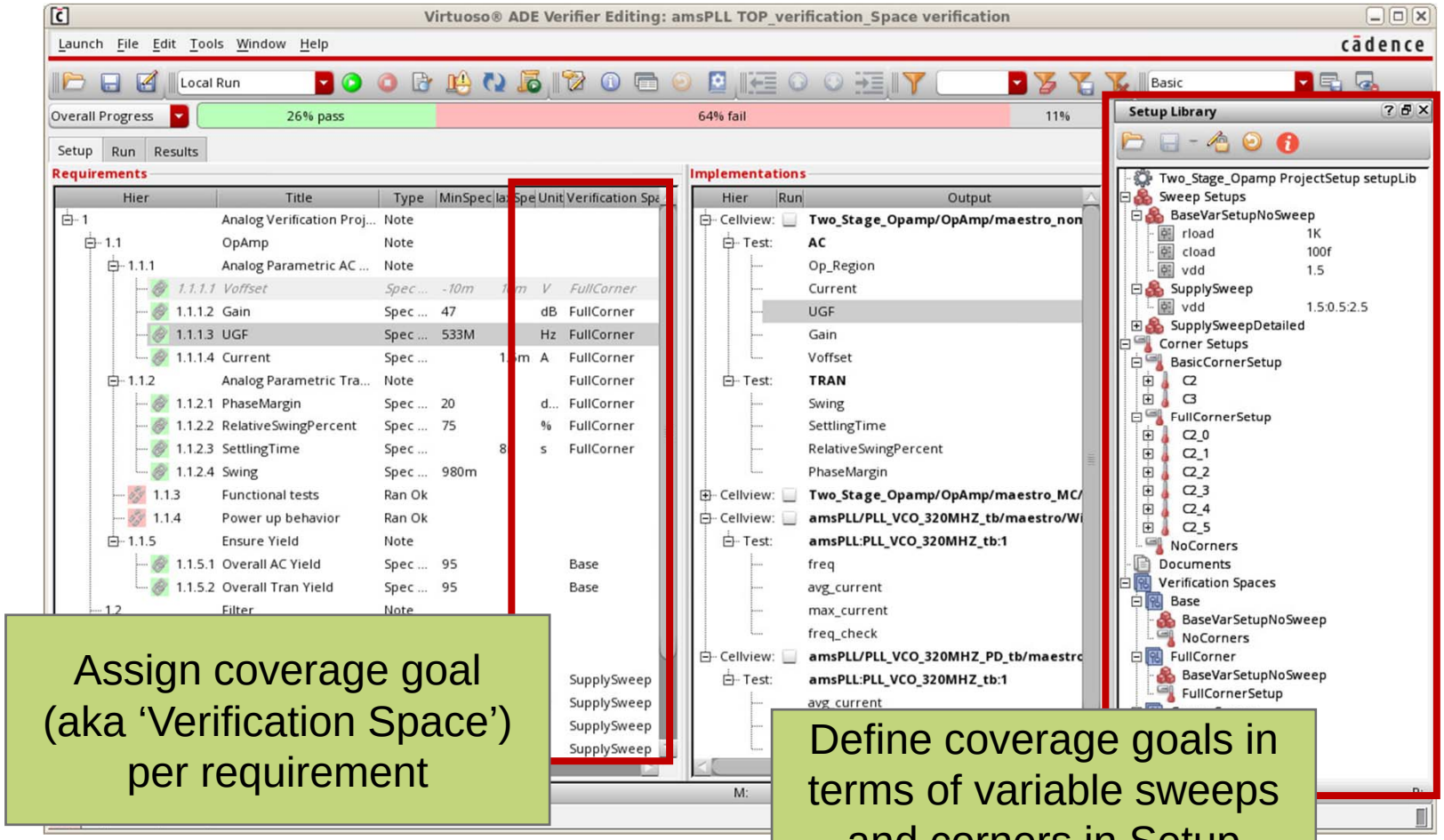
Facilitate multi-user sharing and enables coverage based verification



© 2018 Cadence Design Systems, Inc. All rights reserved.

Virtuoso 6.1.8 including analog coverage support

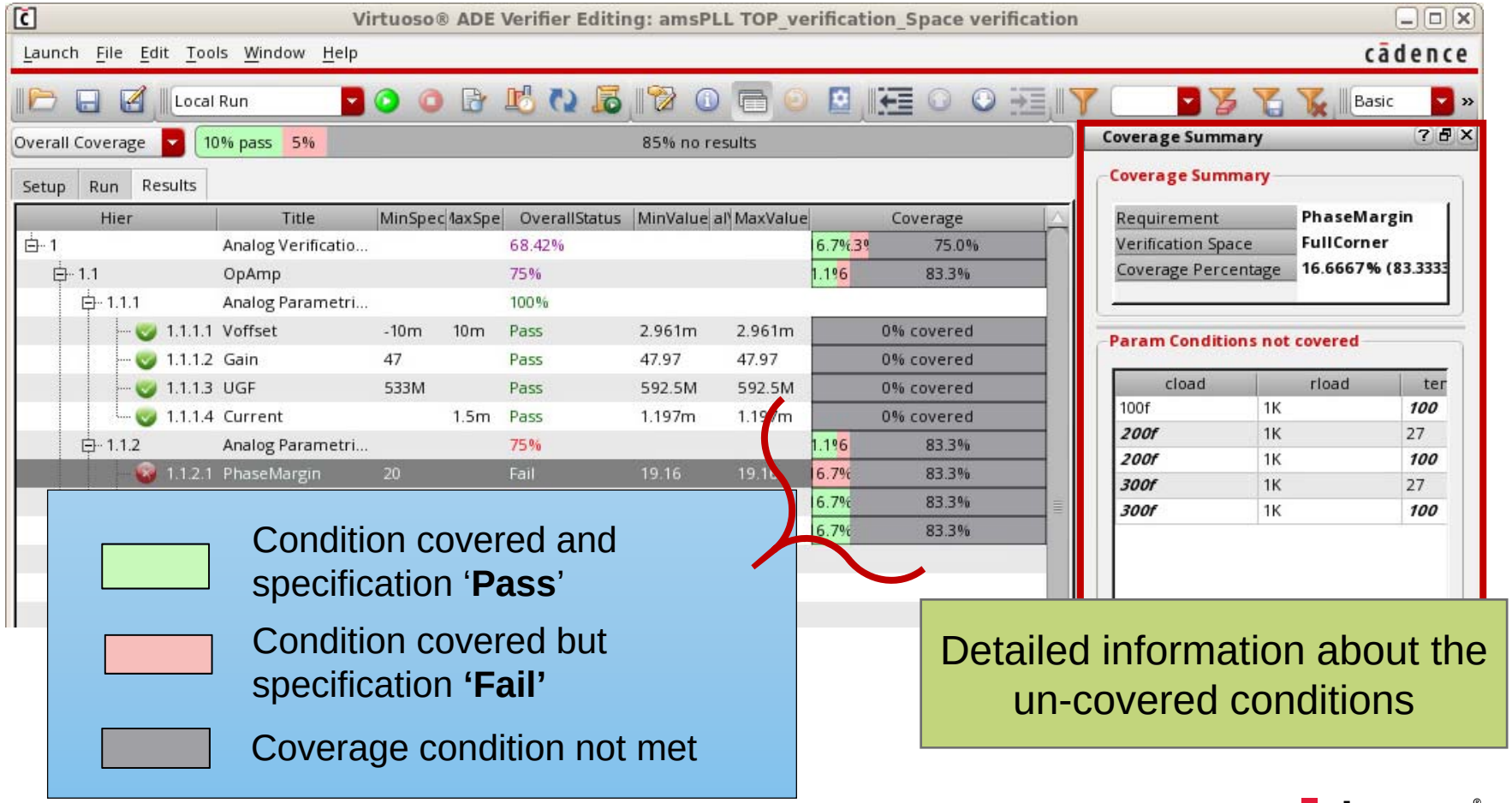
- ADE Assembler / Verifier + Setup Library Assistant



© 2018 Cadence Design Systems, Inc. All rights reserved.
cadence

Monitor analog coverage in Verifier results tab

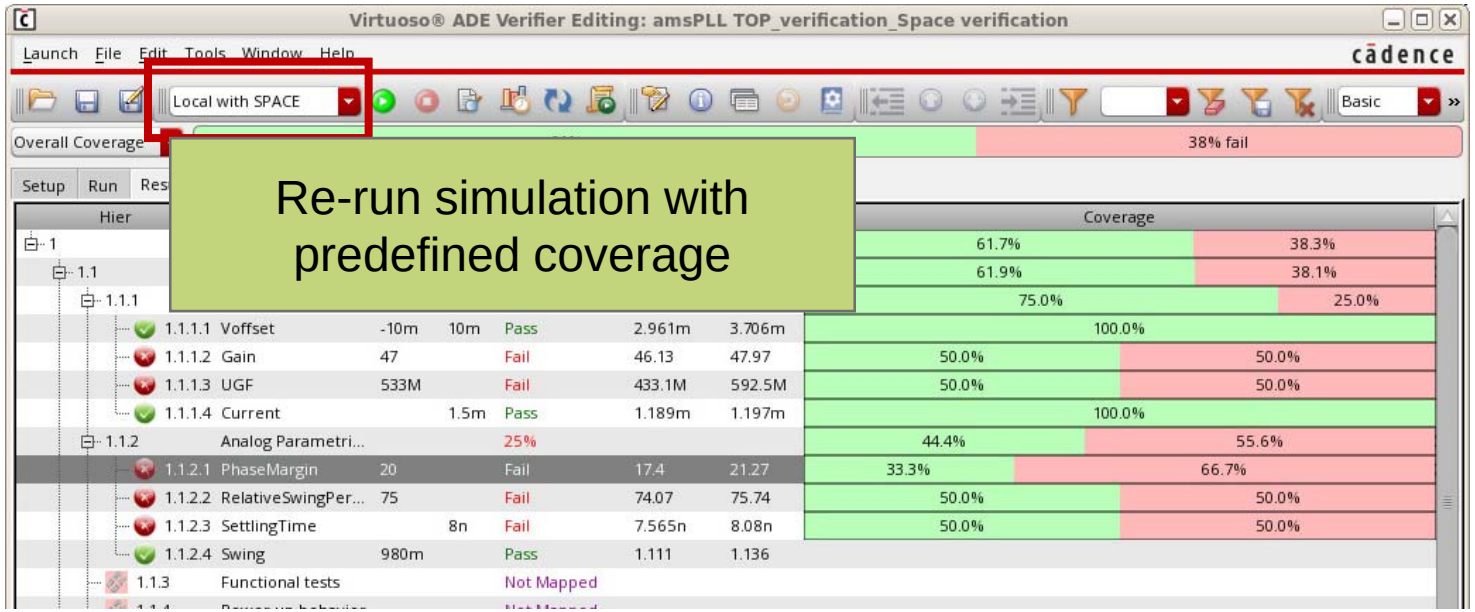
- Results tab shows the analog coverage based on the goals defined above



© 2018 Cadence Design Systems, Inc. All rights reserved.
cadence

Maximizing the coverage

- Coverage provide an accurate verification status report
  - When is the verification process finished?
  - How many simulations are needed to fulfil 100% coverage?
  - Are modifications to the coverage goal needed?
- Verifier allows automatic running of simulation with the predefined coverage goal → resulting in 100% coverage



© 2018 Cadence Design Systems, Inc. All rights reserved.
cadence

Tutorial A:

Analog Coverage – Yesterday‘s Dreams, Today‘s Reality and Tomorrow‘s Capabilities

Coverage Model

Walter Hartong, Lars Hedrich, Markus Olbrich

Background

- Cluster research project ANCONA
  - Coverage
  - UFM and LUH participated as research partners
- Further discussion started at ANALOG 2016 in Bremen

From Verification Tasks to Coverage

- Today
  - Simulation run oriented verification
  - Verification plan assembled by circuit designer
  - What portion of relevant situations does it consider?
  - ➢ Unknown
  - ✓ Coverage can be a solution



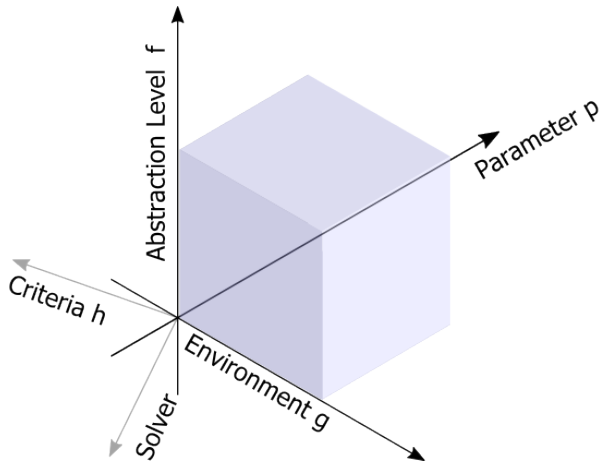
- Open Questions
  - What exactly is the space to cover?
  - What dimensions does it have?
  - How to identify the relevant situations?

Basic Coverage Model (BCM)

- Previously: No analog coverage definition available
- Basic question: What can be covered?
- Idea
  - Systematically identify the multidimensional coverage space
- Approach
  - Start with fundamental circuit model:
    Equation system
  - Examine the equation system components
  - Define dimensions for the components
  - Add additional verification aspects

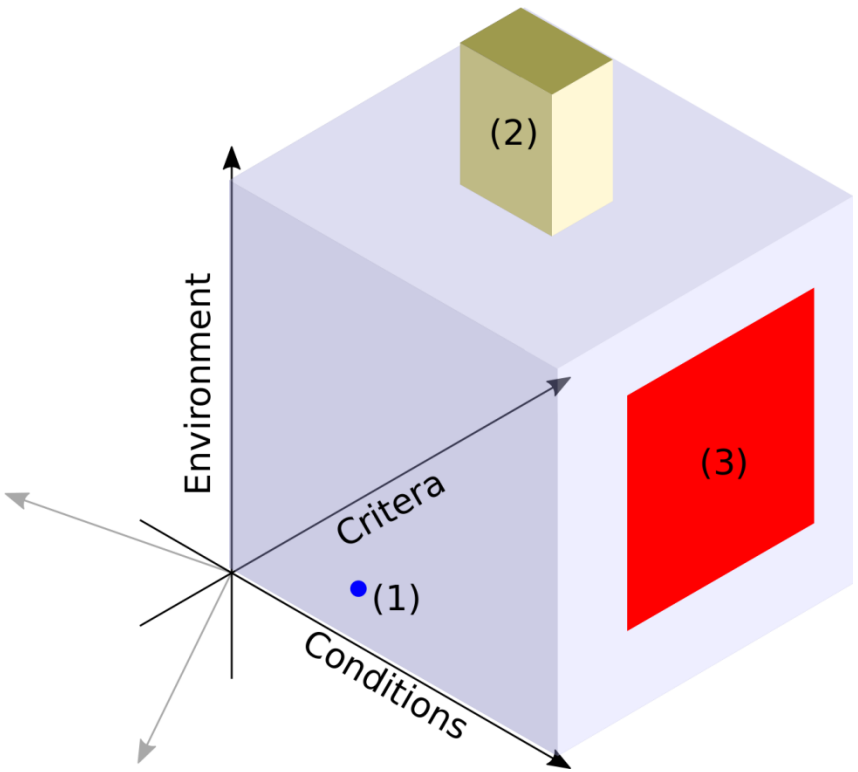
Base Coverage Model (BCM)

1. Model (abstraction level)
2. Environment and Excitation
3. Condition
   a) PVT parameters
   b) Initial states
4. Criteria
   a) Measurement
   b) Output
5. Solver



➢ Each dimension spans a space that can be covered

Coverage Subspaces

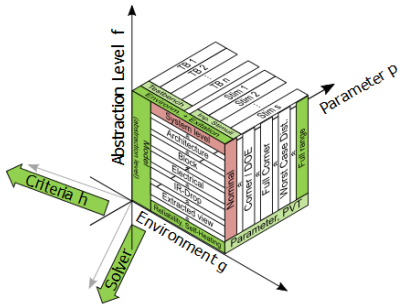


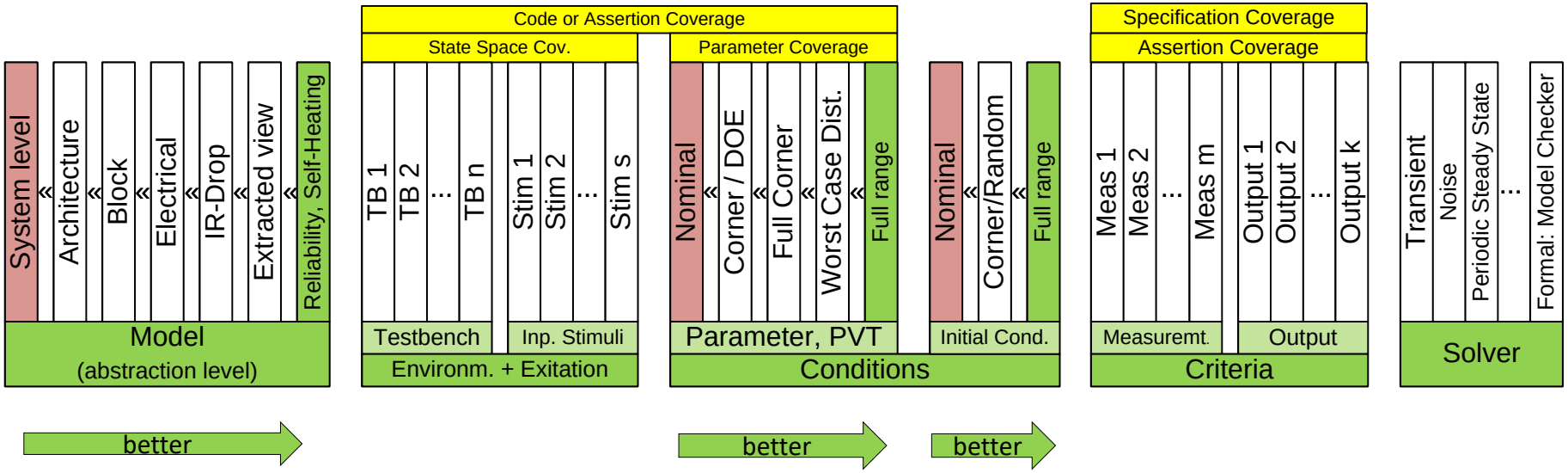
- Multi dimensional subspaces
- Subspaces can be
  - finite (points) (1)
  - infinite (ranges) (2), (3)
- Each verification task (e.g. a simulation) covers a subspace

Coverage Measures

- Necessary: Explicitly defined relevant subspace (coverage goal) (can barely be automated)
- Total Coverage
  - In case of discrete point sets
    $$C = \frac{\#\ Examined\ Relevant\ Verification\ Points}{\#\ Relevant\ Verification\ Points}$$
  - In case of continuous space
    $$C = \frac{Examined\ Relevant\ Subspace\ Volume}{Relevant\ Subspace\ Volume}$$
- Many different partial coverages possible
  - Functional specification coverage
  - Parameter coverage
  - Defined by different relevant subspaces

Unrolled BCM





- BCM Coverage Measures
  - Model
  - Environment and Excitation
  - Condition
  - Criteria
  - Solver
- Special Coverage Measures
  - Assertion Coverage
  - Code Coverage
  - State Space Coverage
  - Specification Coverage

Further Agenda

- Classical verification methods, coverage (Lars Hedrich, UFM)
- Industrial applications (Walter Hartong, Cadence)
- Break
- Concrete measures and more examples (Lars Hedrich, UFM)
- Summary (Markus Olbrich, LUH)
- Q&A

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Leibniz Universität Hannover

cadence®

Analog Coverage - yesterday's dreams, today's reality and tommorrow's capabilities

Walter Hartong, Cadence, Munich
Lars Hedrich, University of Frankfurt
Markus Olbrich, University of Hannover

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

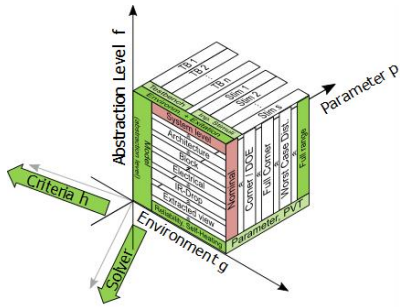
Leibniz Universität Hannover

cadence®

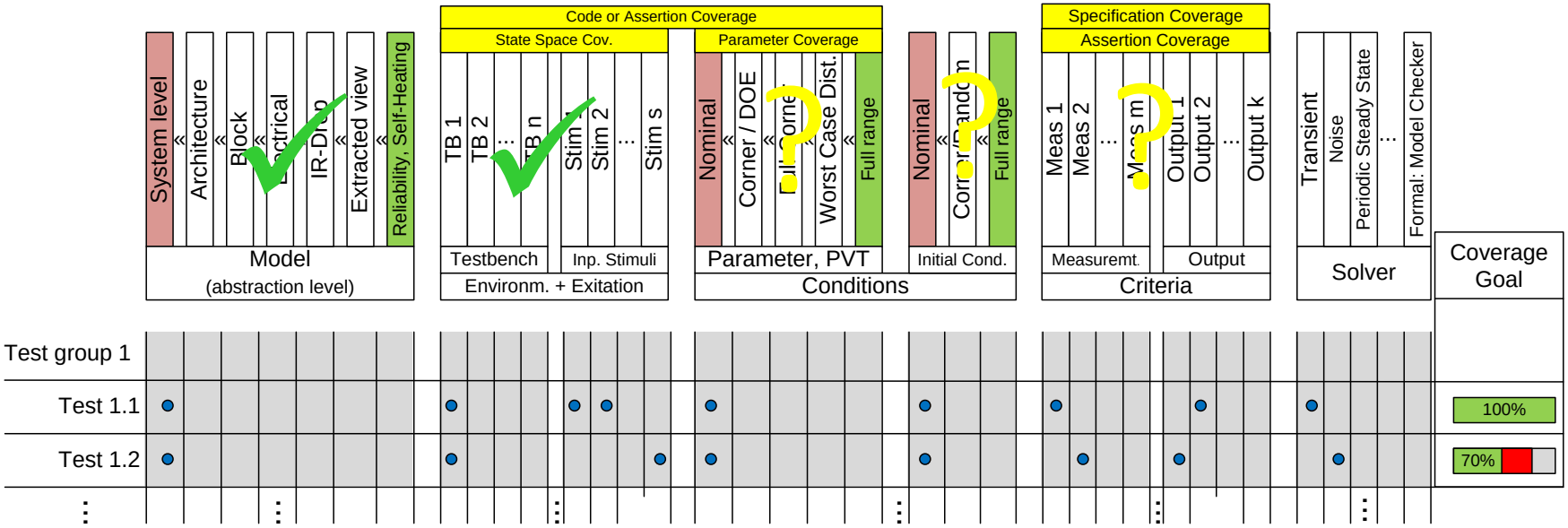
Part II: What next?

Walter Hartong, Cadence, Munich
Lars Hedrich, University of Frankfurt
Markus Olbrich, University of Hannover

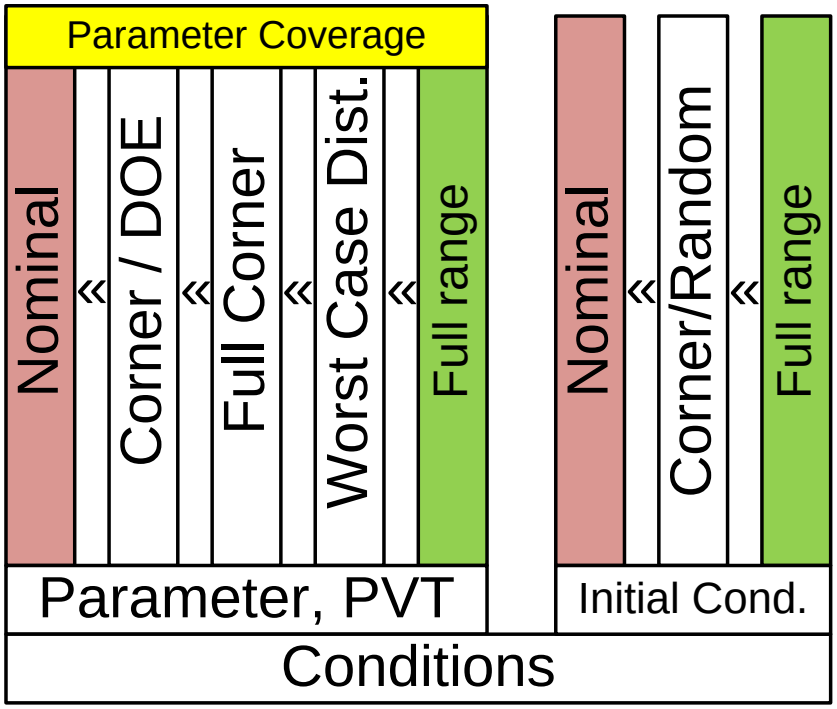
Coverage Control Panel CCP:



- What is available? ✓
- What should be tackled? ?
- Is easy to reach?
- What is needed for strong verification (ISO26262)?

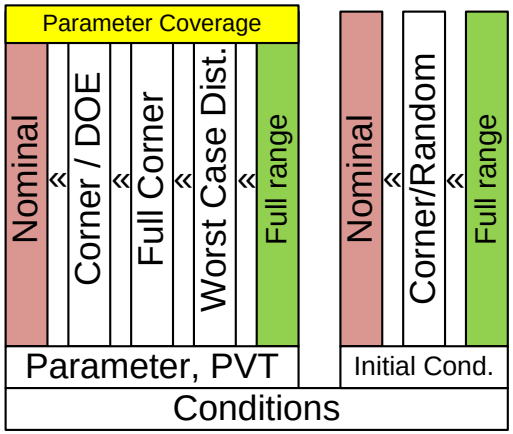


Parameter Coverage



Parameter Coverage II

- Design parameters are sources of false behavior
- High dimensional parameter space:
  - Circuit and process parameters
  - Initial states
  - Operating parameters
  - Environment parameters
  - Temperature
  - Noise
  - …
- Full coverage means check of every parameter



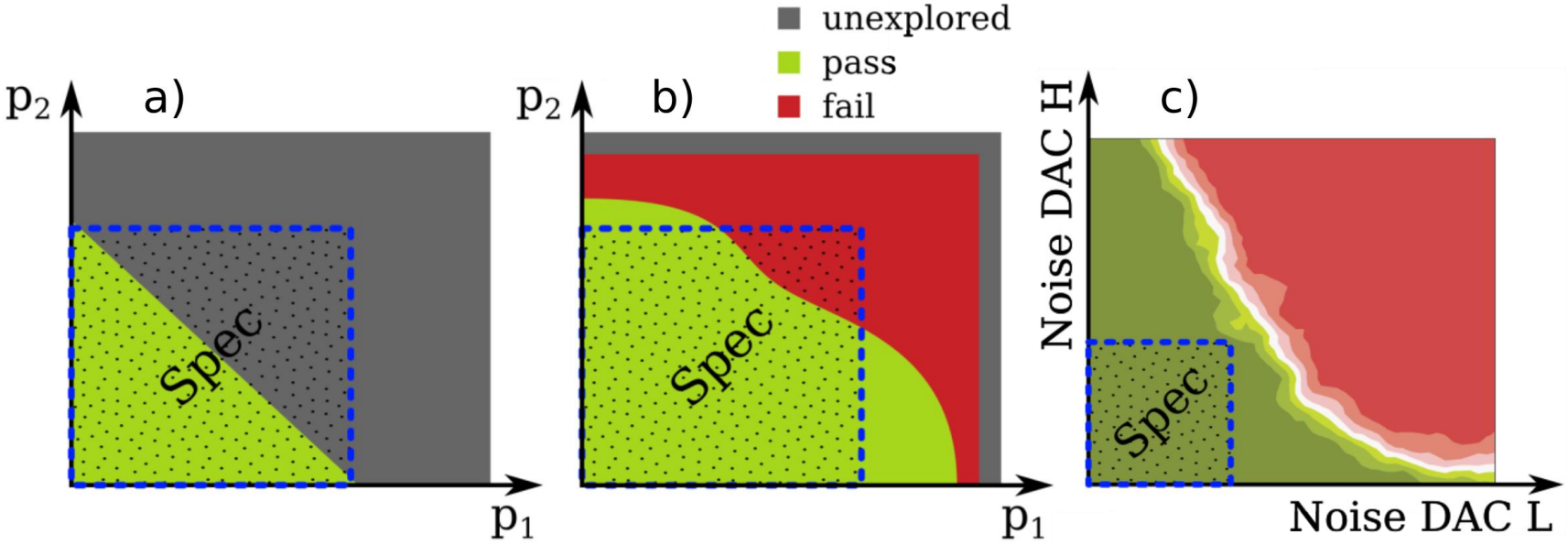
$$\Gamma_{\text{param}} = \frac{\int_{S_p} f_{\text{explored}}(s)ds}{\int_{S_p} ds}$$

- Computational explosion! => First measure discretize

$$Parameter\ Coverage = \frac{\#\ explored\ parameter\ values}{\#\ total\ paramter\ values\ in\ set}$$

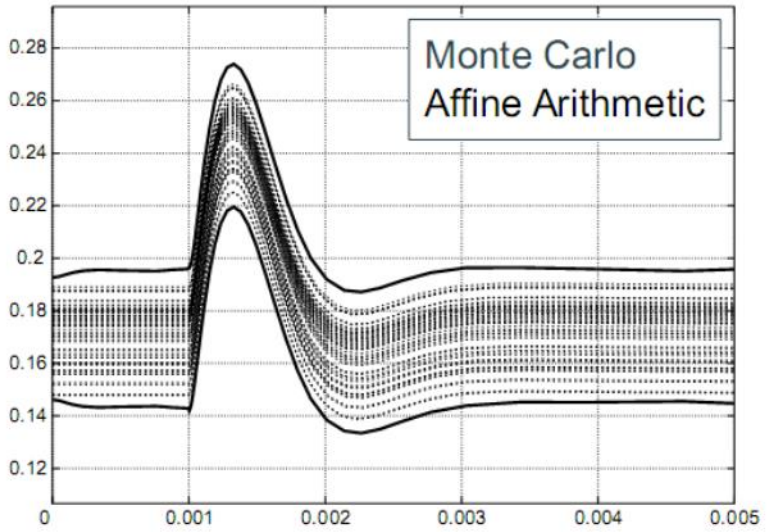
Parameter Coverage III

- Simulation based methods
- Pass / fail behavior of a AMS circuit
- Model non- functional parameters like cross-talk, noice, voltage drops, etc.
- Advanced Methods: Border Search



Other Parameter Oriented Coverage Methods

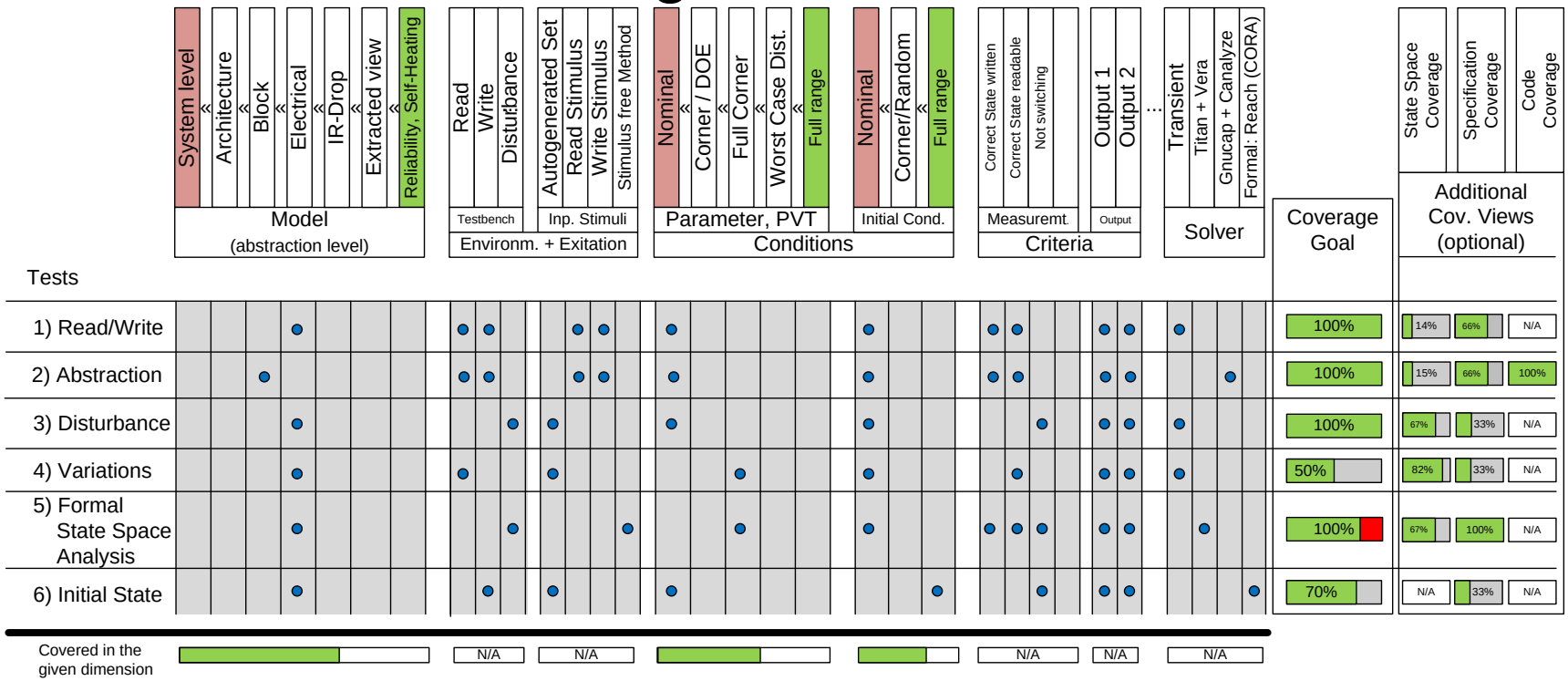
- Monte Carlo
- Worst-Case Distance
- Tail Methods
- Interval / Affine Methods (Transient Simulations)



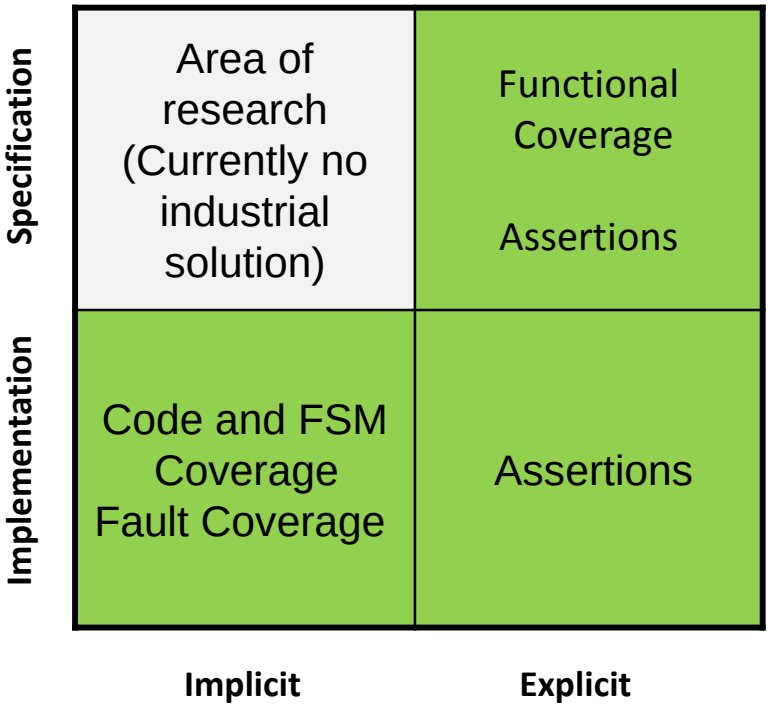
[C.Radojicic, TU Kaiserslautern]

What Else?

Coverage Control Panel



Digital Coverage Methods (not complete)



- Functional Coverage:
  - Test-Cases
  - Assertion Coverage,…
- Code Coverage:
  - Toggle Cov.
  - Line Cov.
  - Statement Cov.
  - Block Cov.
  - Branch Cov.
  - Expression Cov.
- FSM Cov.
- Fault Cov.

- Use a lot of formal methods
- Very complex designs

[Source: Coverage Cookbook from www.verificationacademy.com]

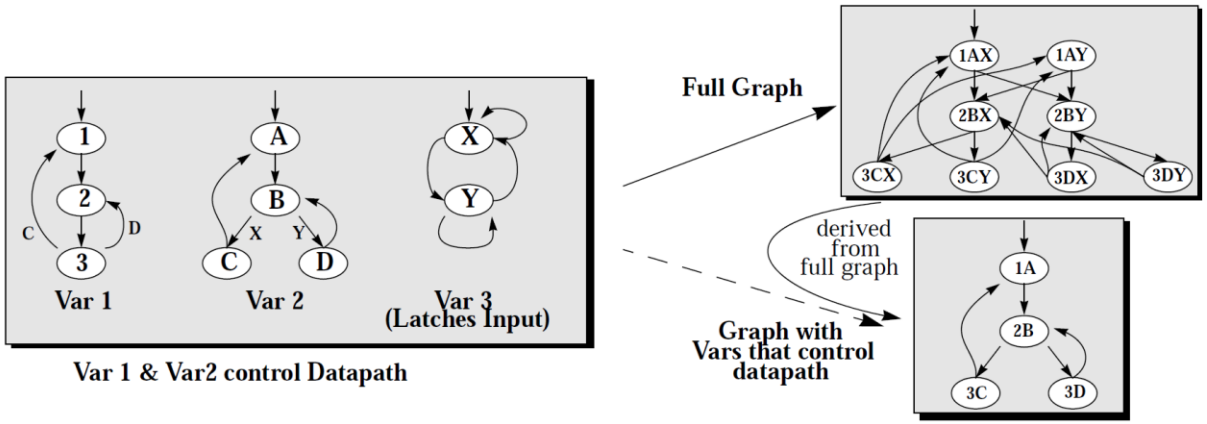
FSM Coverage digital

- Definitions

$$State\ Coverage = \frac{\#\ Visited\ States}{\#\ Total\ Reachable\ States}$$

$$Transistion\ Coverage = \frac{\#\ Taken\ Transitions}{\#\ Total\ Reachable\ Transistions}$$

- Applicable to data path and control path



[Ho, R., Horowitz, M. "Validation coverage analysis for complex digital designs" ICCAD 96]

Code Coverage Digital

- Code Coverage
- Control Coverage
- Data Coverage
- Standard application, commercial available

$$Line\ Coverage = \frac{\#\ visited\ Lines\ of\ Code}{\#\ all\ executable\ Lines}$$

$$Branch\ Coverage = \frac{\#\ visited\ Branches\ of\ (if\ else)}{\#\ all\ Branches}$$

$$Path\ Coverage = \frac{\#\ visited\ \ Path\ through\ (if\ else)}{\#\ all\ Paths}$$

- From formal world, tools to compute vectors to increase

Examples	#Lines	#Tags	Directed			Random		
			#Vec	#OCCOM Coverage%	Line Coverage%	#Vec	OCCOM Coverage%	Line Coverage%
eight queens	220	146	179	87%	92%	2000	67%	90%
dcnew	353	256	338	86%	100%	3000	77%	100%
crd	120	88	156	94%	100%	2000	67%	90%
amp	288	198	168	68%	92%	3000	68%	92%
coherence	530	346	442	85%	94%	5000	54%	90%

[Farzan Fallah "OCCOM-efficient computation of observability-based code coverage metrics for functional verification" 2001]
[Cristian Cadar, et. al. "KLEE: Unassisted and Automatic Generation of High-Coverage Tests for Complex Systems Programs" 2008]
[Srinivas Devadas et.al. "An observability-based code coverage metric for functional simulation" 1997]

Part V
Advanced Coverage: Low Hanging Fruits

Code Coverage: Analog

- For Verilog-A Behavioral Models
- Path coverage: Problem false paths
- Gnucap and ADMSXml – based full automatic approach
- Spectre-Add-On available

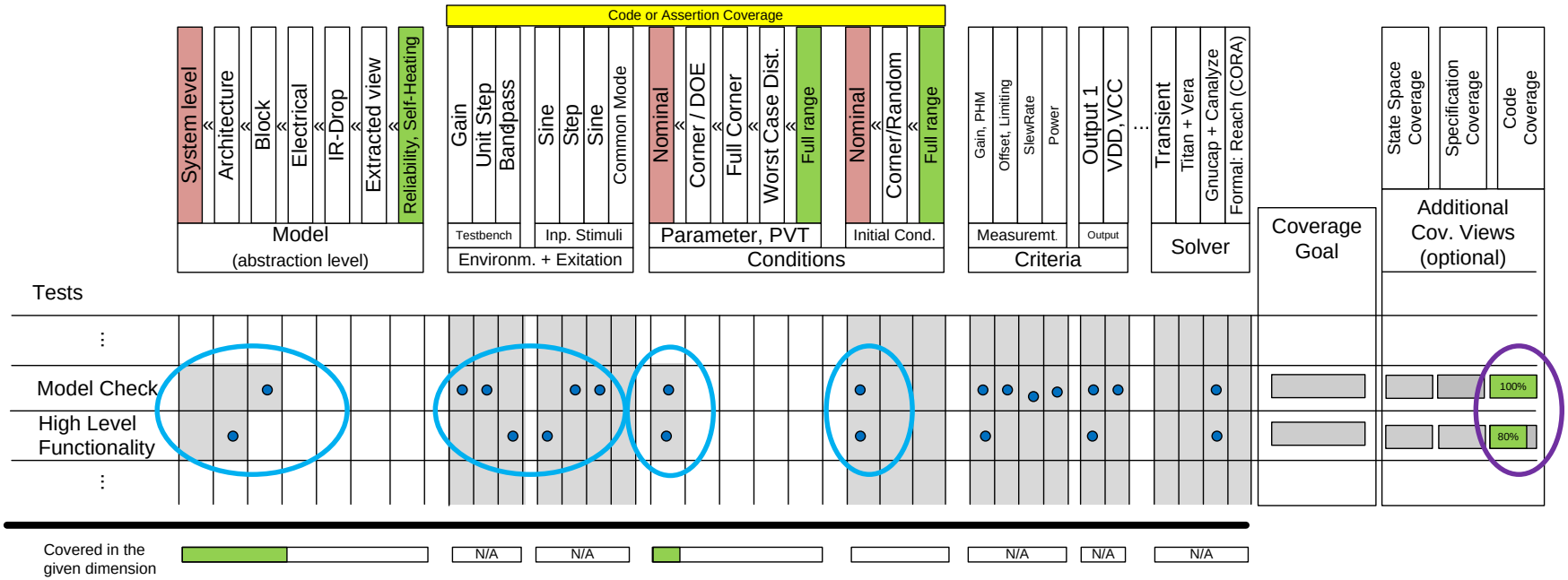
$$Line\ Coverage = \frac{\#\ visited\ Lines\ of\ Code}{\#\ all\ executable\ Lines}$$

$$Branch\ Coverage = \frac{\#\ visited\ Branches\ of\ (if\ else)}{\#\ all\ Branches}$$

$$Path\ Coverage = \frac{\#\ visited\ \ Path\ through\ (if\ else)}{\#\ all\ Paths}$$

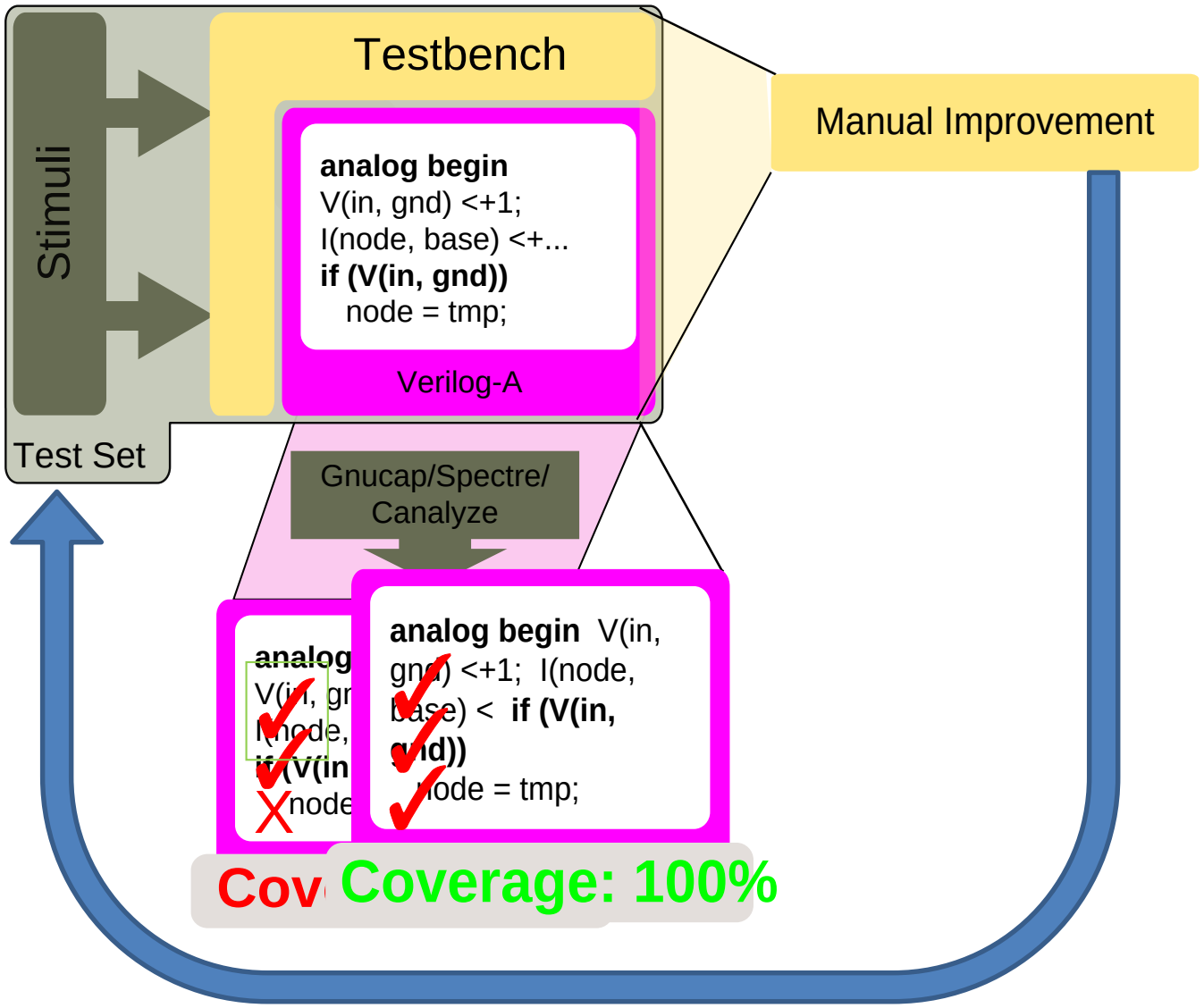
[[Y. Sha, et.al. "On code coverage measurement for Verilog-A", 2004
Andreas Fürtig et.al. “Comparing code coverage metrics for analog behavioral models" 2017]

Defining a Coverage Goal for Code Coverage for an Behavioral Model of an OP



- Define Test Cases and Coverage in known Dimensions
- Define Coverage Goal

Code Coverage Methodology



Hands-on Session Code Coverage

- Example: Op / Test-Chip

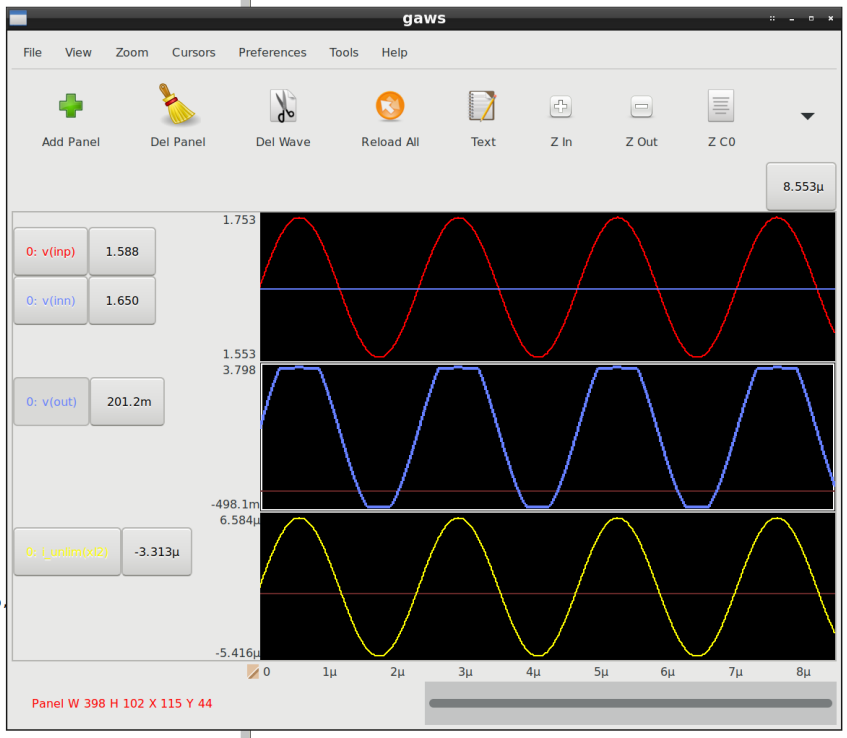
- Use for Model-Sign-Off / Equivalence Checking
- Simple: Find missing Stimuli / Testbench
- Advanced: Generate missing Stimuli

  – COVA Tool Download:
  http://www.em.cs.uni-frankfurt.de/index.php?id=218

Hands-On Session: OP First If statements covered by Input stimuli

```
 81  o |       //        I(gnd, za) <+ 0 * (V(inp, inn)+voff);
 82  o |
 83  o |       // slew rate limiting
 84  ✓ |       i_unlim = gm1* (V(inp, inn)+voff);
 85  o |       //$strobe(" i_unlim %g imax %g ",i_unlim, imax);
 86  ✓ |       I(z1,gnd) <+ i_unlim;
           |
 87  o | |       if (i_unlim > imax)
 88  o | |         begin
 89  × | |           I(z1,gnd) <+ -1*i_unlim;
 90  o | |         end
 91  o | |       else
 92  o | |         begin
           |
 93  o | |           if (i_unlim < -imax)
 94  o | |             begin
 95  × | |               I(z1,gnd) <+ -1*i_unlim;
 96  o | |             end
           | |                 ELSE MISSING!
 97  o | |         end
           |
 98  o |
 99  o |       // gm to ground
100  ✓ |       I(z1,gnd) <+ gm1*V(z1,gnd);
101  o |       // Cap to second node=compensation
102  ✓ |       I(za, z1) <+ C1 *ddt(V(za, z1));
103  o |
104  o |       // gain Stage
105  ✓ |       I(za, gnd) <+ gm2*(V(za, gnd)+gain*V(z1,gnd));
106  o |
107  o |       // finite output swing
           |
108  o | |       if( V(za,gnd) > output_range/2 )
109  o | |         begin
110  ✓ | |           I(za, gnd) <+ (V(za, gnd)-output_range/2) / Rf;
111  o | |         end
           | |                 ELSE MISSING!
           |
112  o | |       if( V(za,gnd) < -output_range/2 )
113  o | |         begin
114  ✓ | |           I(za, gnd) <+ (V(za, gnd)+output_range/2) / Rf;
115  o | |         end
           | |                 ELSE MISSING!
116  o |       // output stage
117  ✓ |       vvout = V(za, gnd)+output_offset;
118  o |       //$strobe(" vdiffin %g za %g vvout %g offset %g ",V(inp,
put_offset);
119  o |       // output with gm2
120  ✓ |       I(gnd, out1) <+ gm2 * vvout;
121  ✓ |       I(gnd, out1) <+ V(gnd, out1) * gm2;
122  o |       // Vbias input
123  ✓ |       I(vbias3,gnd) <+ V(vbias3, gnd)/Rf;
124  o |     end
125  o | endmodule
126  o |
-----------------------------------------------
SUMMERY (opbeh.va):
126 lines in total, 107 ignored
19 codelines, 17 covered by a test, 2 were not
4 ifs, from which 1 have an else branch
8 branches, 2 covered, 6 not reached
Overall Statement Coverage: 89.47%
Branch Coverage:            25.00%
```

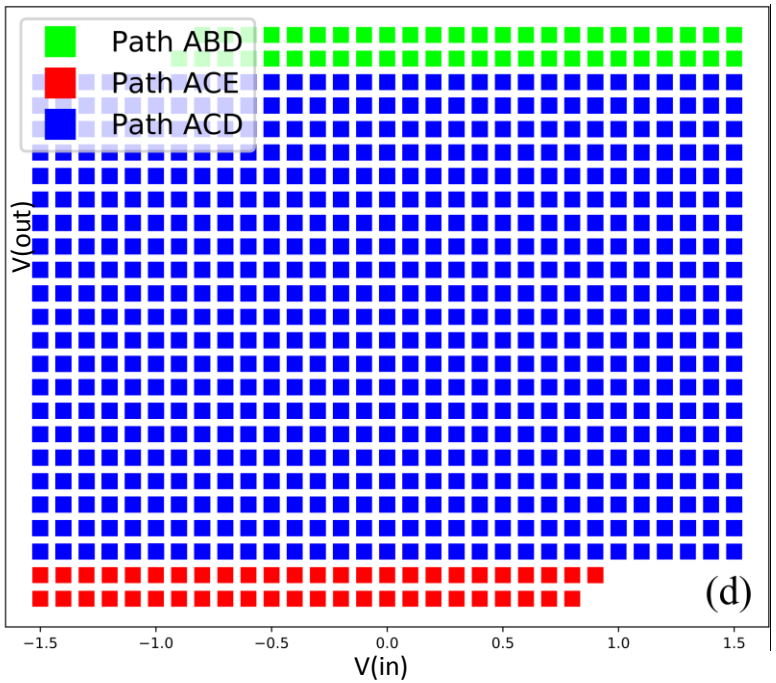
```
h>
```



Code Coverage Simple Example

```
14 ⊙ |    parameter real c=1e-6 from [0:inf);
15 ⊙ |    parameter real r=1e3 from [0:inf);
16 ⊙ |    analog begin
17 ⊙ |
18 ✓ |      I(in, out) <+ V(in, out) / r;
19 ✓ |      I(out, g) <+ c*ddt(V(out, g));
20 ⊙ |
   ⊙ |
21 ⊙ |      if(V(out, g) > 1.0) begin
22 ✓ |        I(out, g) <+ ((V(out, g)-1.0)
23 ⊙ |       end
   ✓ |                          ELSE MISSING!
24 ⊙ |
   ⊙ |
25 ⊙ |      if(V(out, g) < -1.0) begin
26 × |        I(out, g) <+ ((V(out, g)+1.0)
27 ⊙ |       end
   ✓ |                          ELSE MISSING!
28 ⊙ |
29 ⊙ |    end
30 ⊙ |
31 ⊙ | endmodule
```

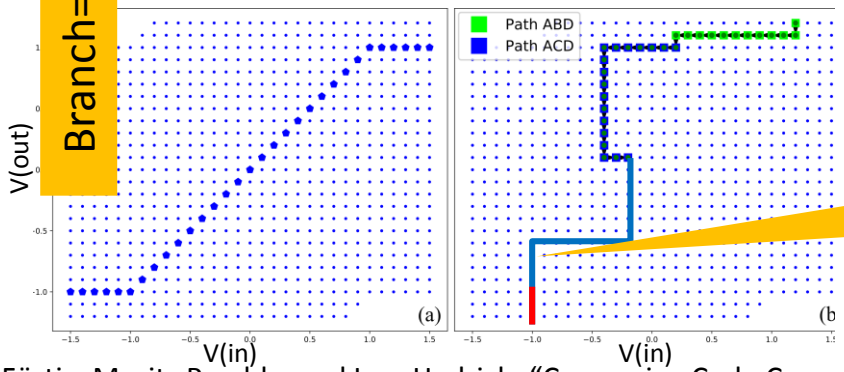
(c)



(d)

Line=83%

Branch=75%

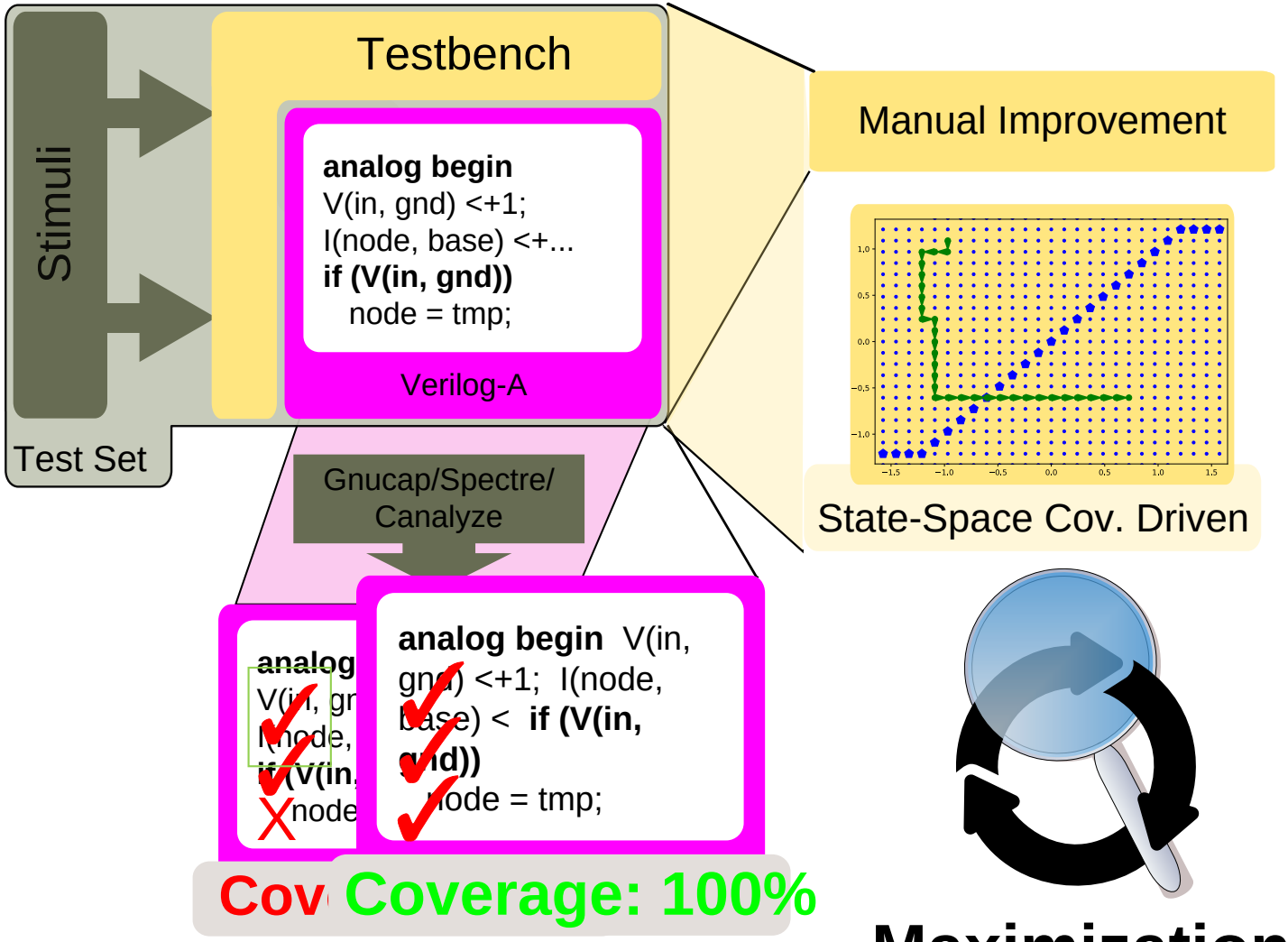


Line 100%
Branch 100%



Andreas Fürtig, Moritz Paschke and Lars Hedrich, “Comparing Code Coverage Metrics for Analog Behavioral Models”, To appear in SMACD, Taormina, Italien, 2017

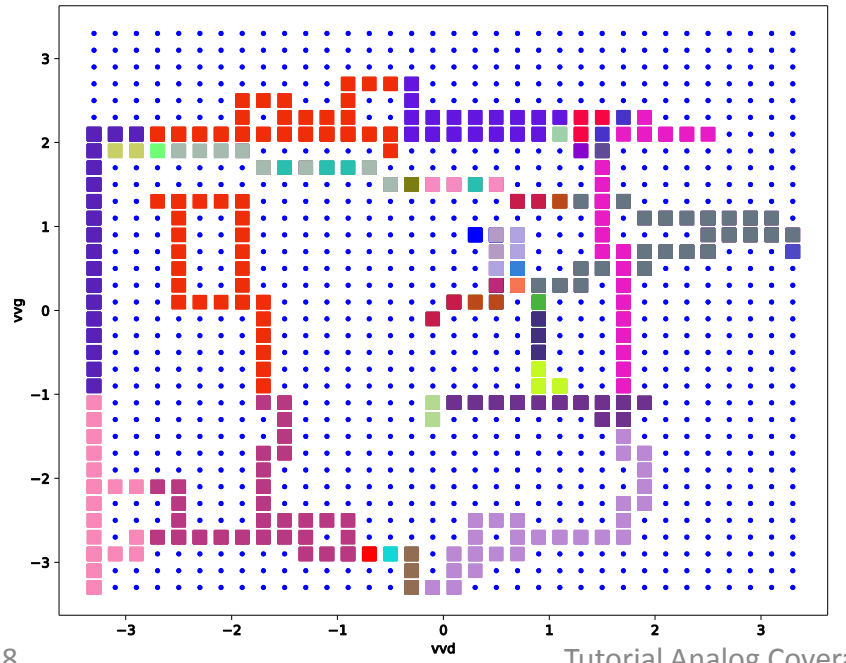
Improving Code Coverage by State-Space Driven Methods

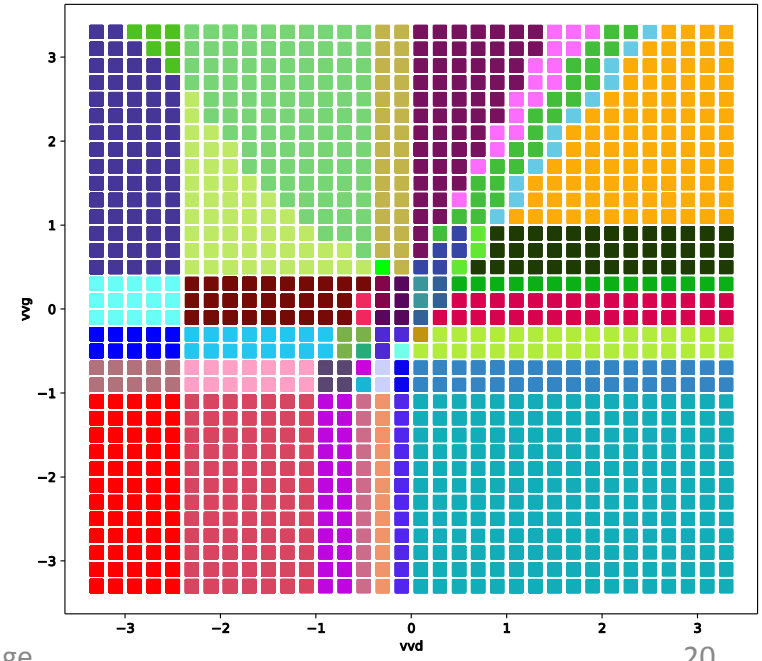


Code Coverage: Complex Example: EKV Model

- **750 Lines of Code**
- **5 Testbenches / Parametersets needed**
- **100% Coverage reached**
- **Many Paths**

```
632 ✓ |          qst = coxt*vt*qs;
633 ✓ |          qgt = coxt*vt*qg;
634 ✓ |          qbt = coxt*vt*qb;
635 o |          //$strobe("qdt = %e qd=%f qst = %e \n",qdt,qd, qst);
636 ✓ |          cbs = 0;
637 ✓ |          cbd = 0;
638 o |
639 o |          if (cbs0>0) begin
640 o |            if (`MTYPE*V(b,si)>`FC*`PB)
641 x |              cbs = cbs+cbs0/pow(1-`FC,MJ)*(1+MJ*(`MTYPE*V(b,si)-`PB*`FC))/(`PB*(1-`FC));
642 o |            else
643 ✓ |              cbs = cbs+cbs0/pow(1-`MTYPE*V(b,si),MJ);
644 o |          end else begin
645 x |            tmp4=56;
646 o |          end
647 o |          if (cbd0>0) begin
648 o |            if (`MTYPE*V(b,di)>`FC*`PB)
649 x |              cbd = cbd+cbd0/pow(1-`FC,MJ)*(1+MJ*(`MTYPE*V(b,di)-`PB*`FC))/(`PB*(1-`FC));
650 o |            else
651 ✓ |              cbd = cbd+cbd0/pow(1-`MTYPE*V(b,di),MJ);
652 o |          end else begin
653 x |            tmp4=56;
654 o |          end
655 o |          if (cbs0sw>0) begin
656 o |            if (`MTYPE*V(b,si)>FCSW*PBSW)
```





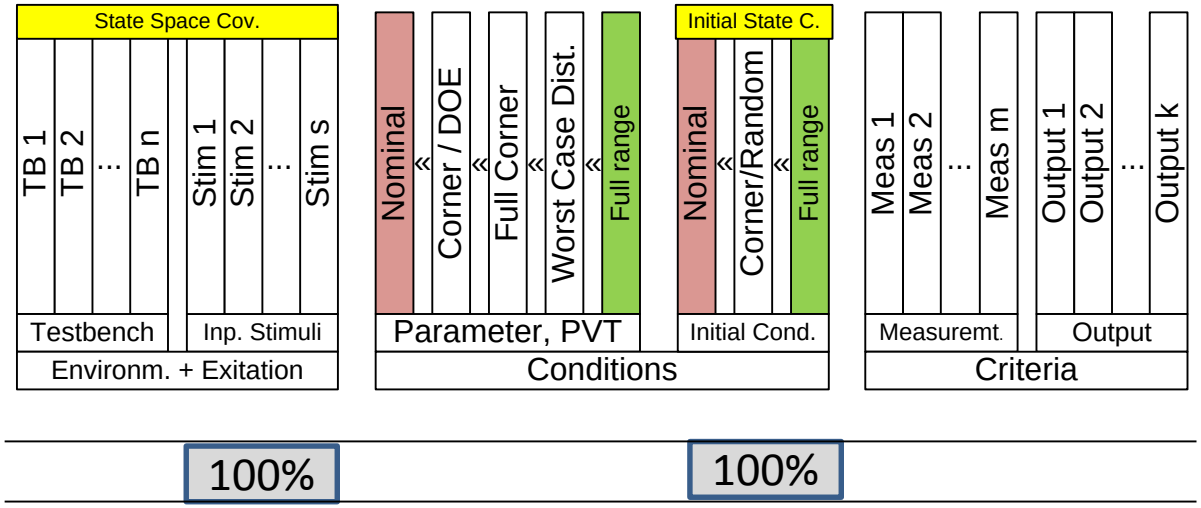
GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Leibniz Universität Hannover

cadence®

Part VI: Tommorrow's Capabilities

State Space Coverage

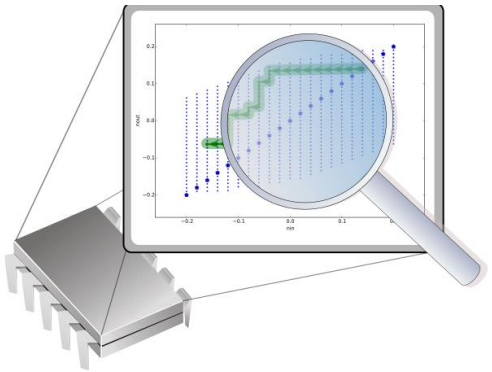


- Goal indirect definable!
- Motivating Examples:
  - Level Shifter : Hanging State
  - Sigma-Delta: Integrator into limits

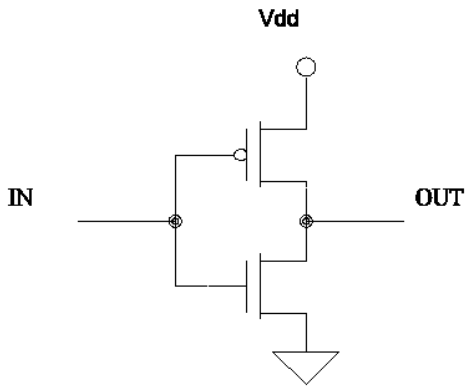
State Space Coverage

- Idea:
  - Judge verification concerning visited states
  - Some discretization mandatory
  - Find a metric to rate a simulation response

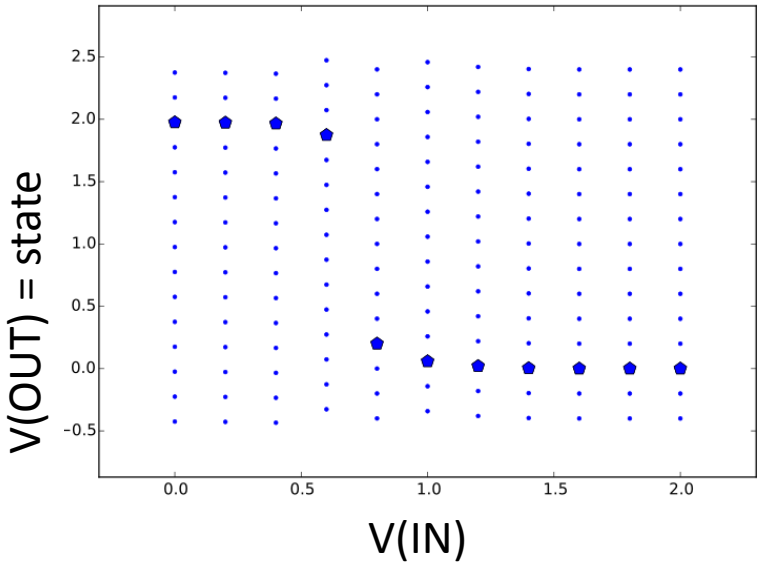
$$State\ Coverage = \frac{\#\ visited\ \ States}{\#\ all\ States}$$

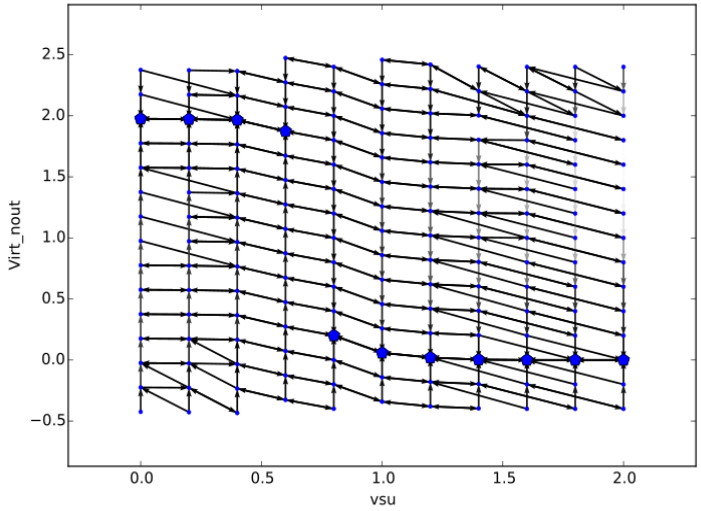


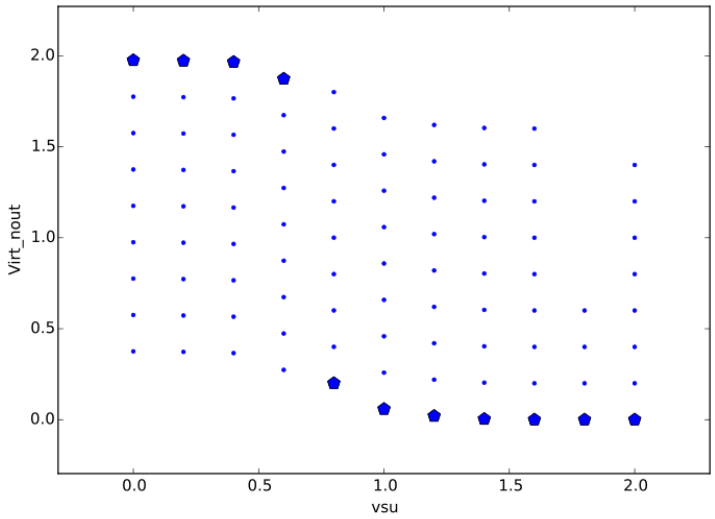
State Space Coverage – Discretization



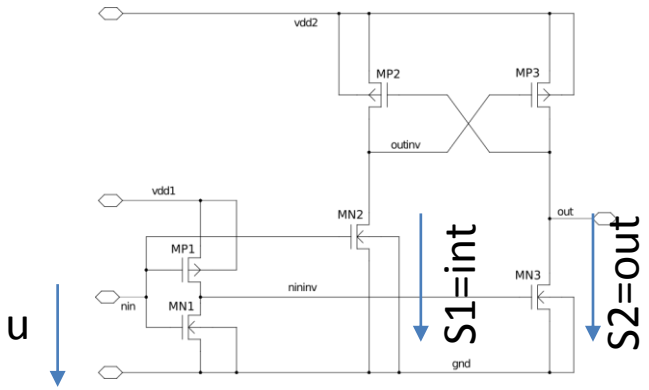
Inverter circuit

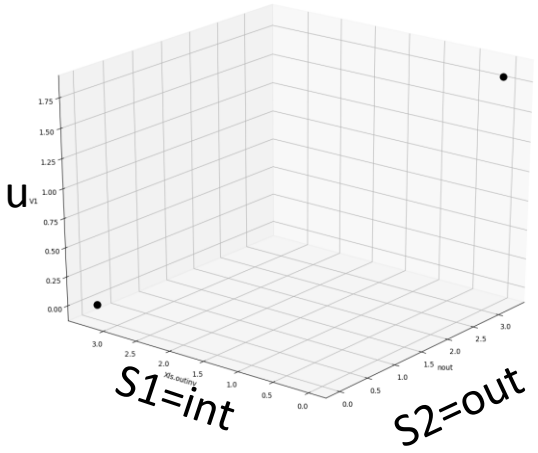


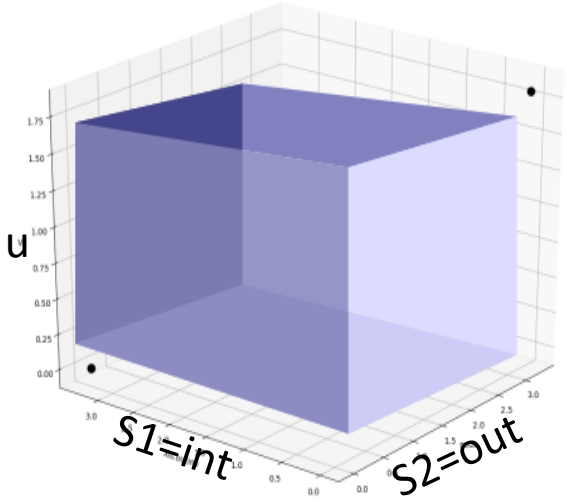




Levelshifter: State Space Coverage





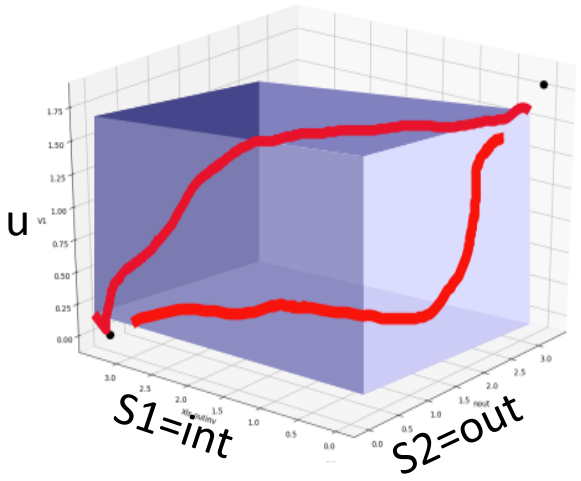


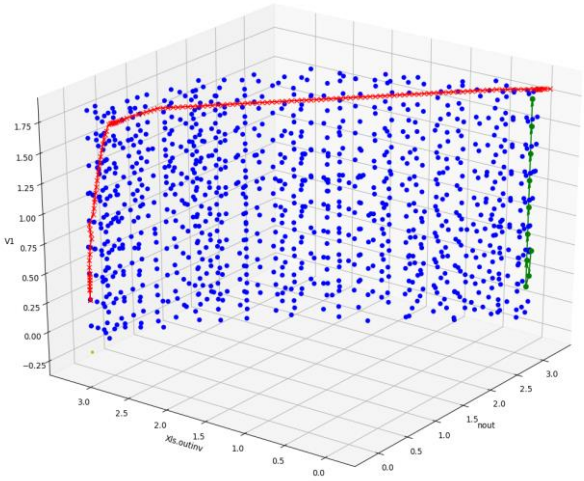
- 2 states, 1 input
- 2 DC-Endpoints

- Coverage goal:
  0..1.8 u x
  0..3.3 s1 x
  0..3.3 s1

=> Box

Reaching the Coverage Goal





- First Simulation
- Discretize space

$\Rightarrow$ 2% Coverage

$\Rightarrow$ 4% Coverage

100 Sim. => 18% Cov.

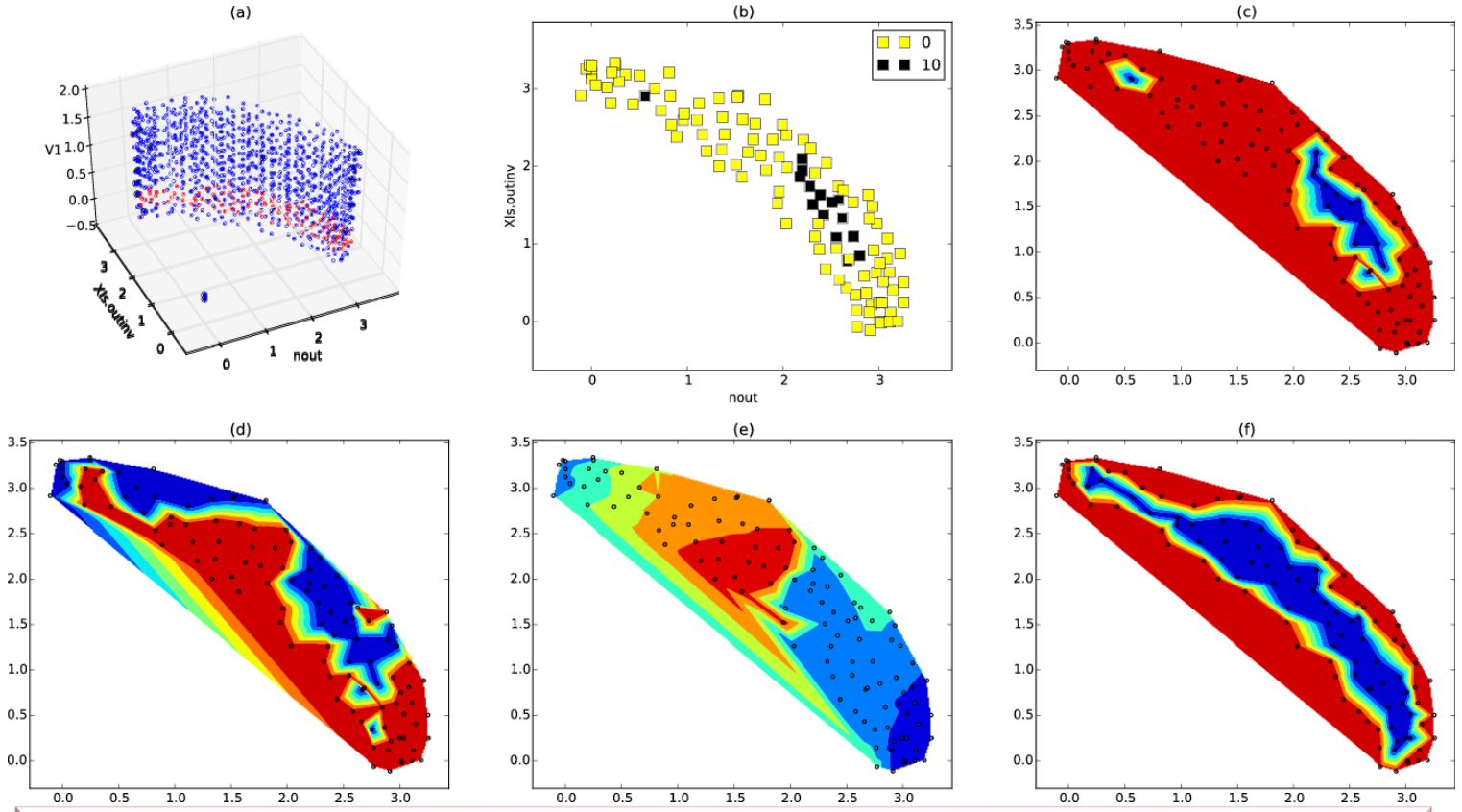
- Denominator wrong =>
- Use reachable region
  - Manually
  - Formal Tools

=> 43% Cov.

$$State\ Coverage = \frac{\#\ visited\ \ States}{\#\ all\ States}$$

State Space Coverage – Do not visit all points

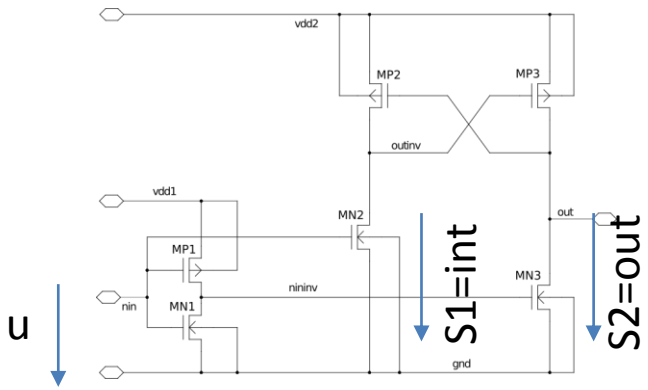
Example: Levelshifter

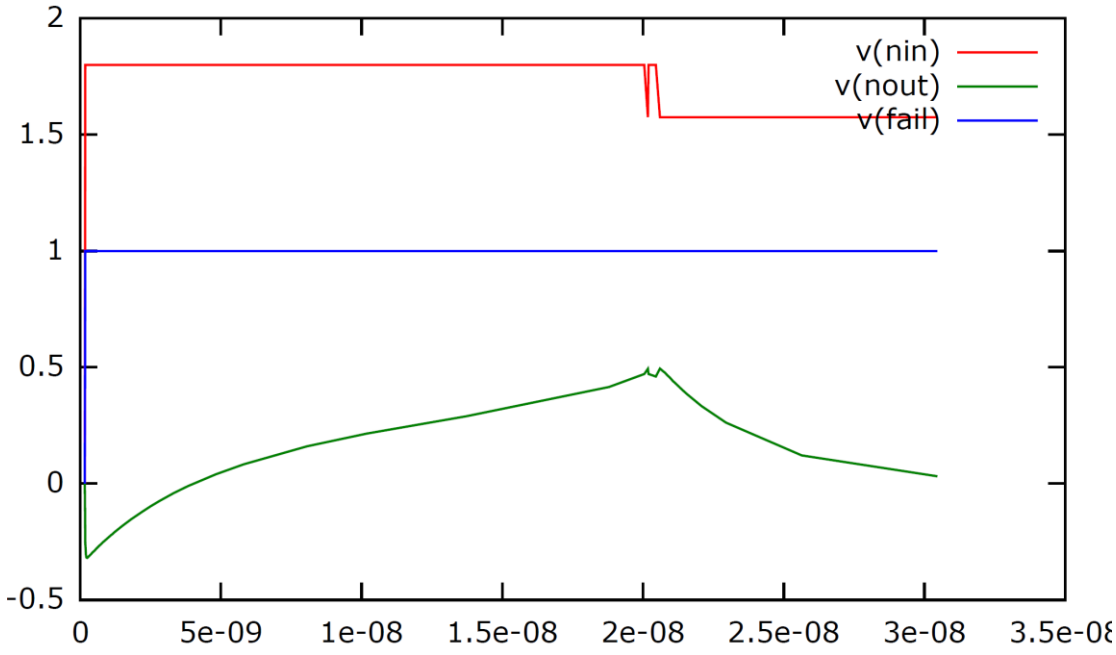


$$\lambda - \text{State Space Coverage} = \frac{\#\,\text{visited Points (Stimuli)}}{\#\,\text{Points in nonlinear, reachable Regions}}$$

a) 3D State Space b) weights for path search, c) weight of state space analysis
d) Dynamical regions e) Eigenvalues f) Border regions (V1 == 0.0V)

Levelshifter: Failure in Behavior





State-Space Coverage & - Optimization: Results

- Various results from project:
  - Optimizing stimuli to find maximum coverage!
  - Proposed = λ-Coverage
  - Normal = Simple state-space coverage
  - Single = Old approach (one, large single stimulus)

Analog Circuit	Method	Coverage > 50% Simulations	Coverage > 50% WCT	Coverage > 75% Simulations	Coverage > 75% WCT	Overall Coverage in % Coverage	Overall Coverage in % Simulations	Overall Coverage in % WCT
Lowpass Filter	*normal*	1	6.22	2	7.38	ζ 100.00	6	9.2
	proposed	1	3.07	2	3.52	ζ_λ 100.00	7	7.75
	single	—	—	—	—	ζ 97.33	1	12.1
Schmitt Trigger	*normal*	3	98.96	5	134.06	ζ 87.06	17	215.83
	proposed	1	5.62	2	8.16	ζ_λ 91.40	3	10.79
	single	—	—	—	—	ζ 30.32	1	19.29
Bandpass Filter	*normal*	6	74.87	15	124.95	ζ 83.06	173	556.86
	proposed	1	31.08	6	97.37	ζ_λ 88.99	18	164.75
	single	—	—	—	—	ζ 62.59	1	388.96
Level-shifter	*normal*	—	—	—	—	ζ 43.91	9	81.46
	proposed	1	12.81	—	—	ζ_λ 72.21	6	27.97
	single	—	—	—	—	ζ 66.39	1	266.24

Initial State Space Coverage II (LUH): „Reachable Set with Guard Intersection“

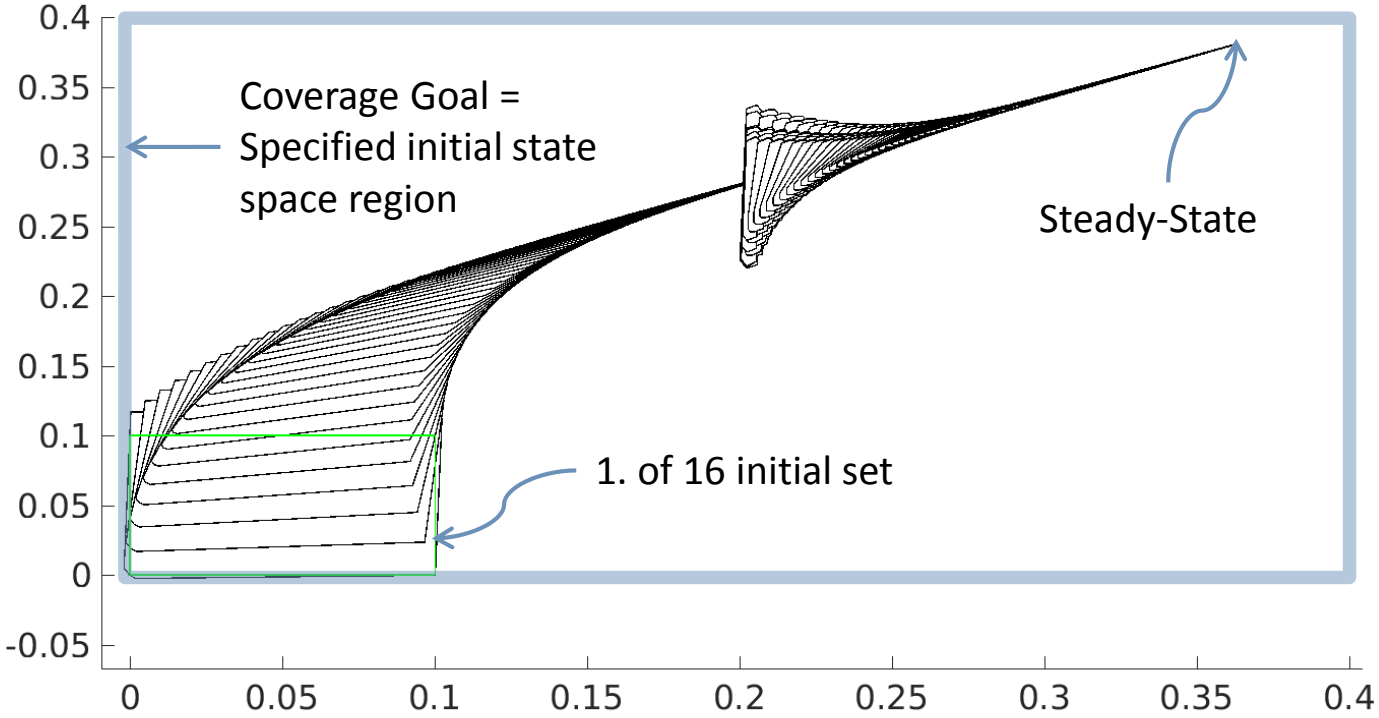
- Idea:

$$\text{Initial State space coverage} = \frac{\text{visited initial state space region (stimuli)}}{\text{specified initial state space region}}$$

- Method:
  - Use sets! (affine arithemtics)
  - On the fly during modeling for a given stimuli
  - Predefined specified initial state space region
- Related Coverage / Method:
  - Affine simulation on system level (TUK)
- Applications:
  - Finding Instabilities, stuck states for oscillators

Reachable Set with Guard Intersection

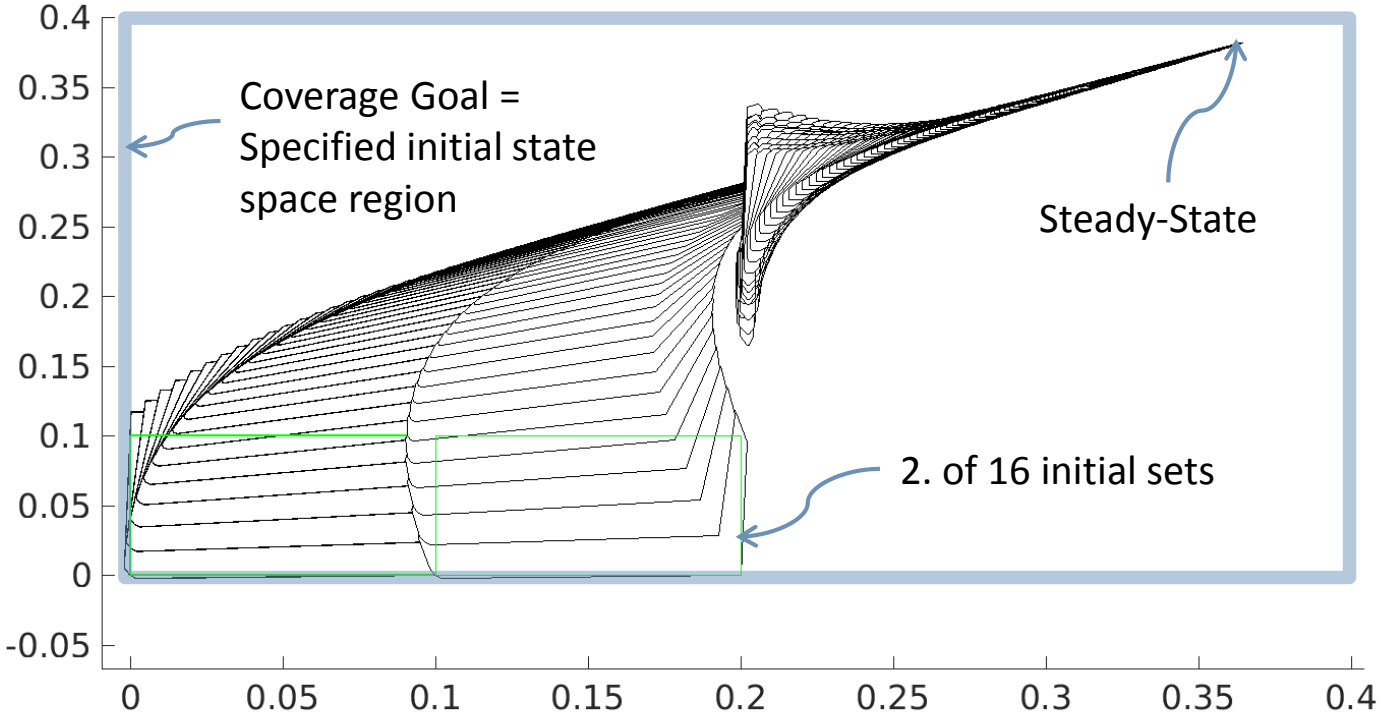
› **Run: 1 von 16**



$$\text{Initial State space coverage} = \frac{\text{visited initial state space region (stimuli)}}{\text{specified initial state space region}} = 6.25\%$$

Reachable Set with Guard Intersection

› **Run: 2 von 16**

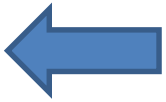


$$\text{Initial State space coverage} = \frac{\text{visited initial state space region (stimuli)}}{\text{specified initial state space region}} = 12.5\%$$

=> Could converge to 100%

Code-Coverage and State-(Initial)-Space Coverage

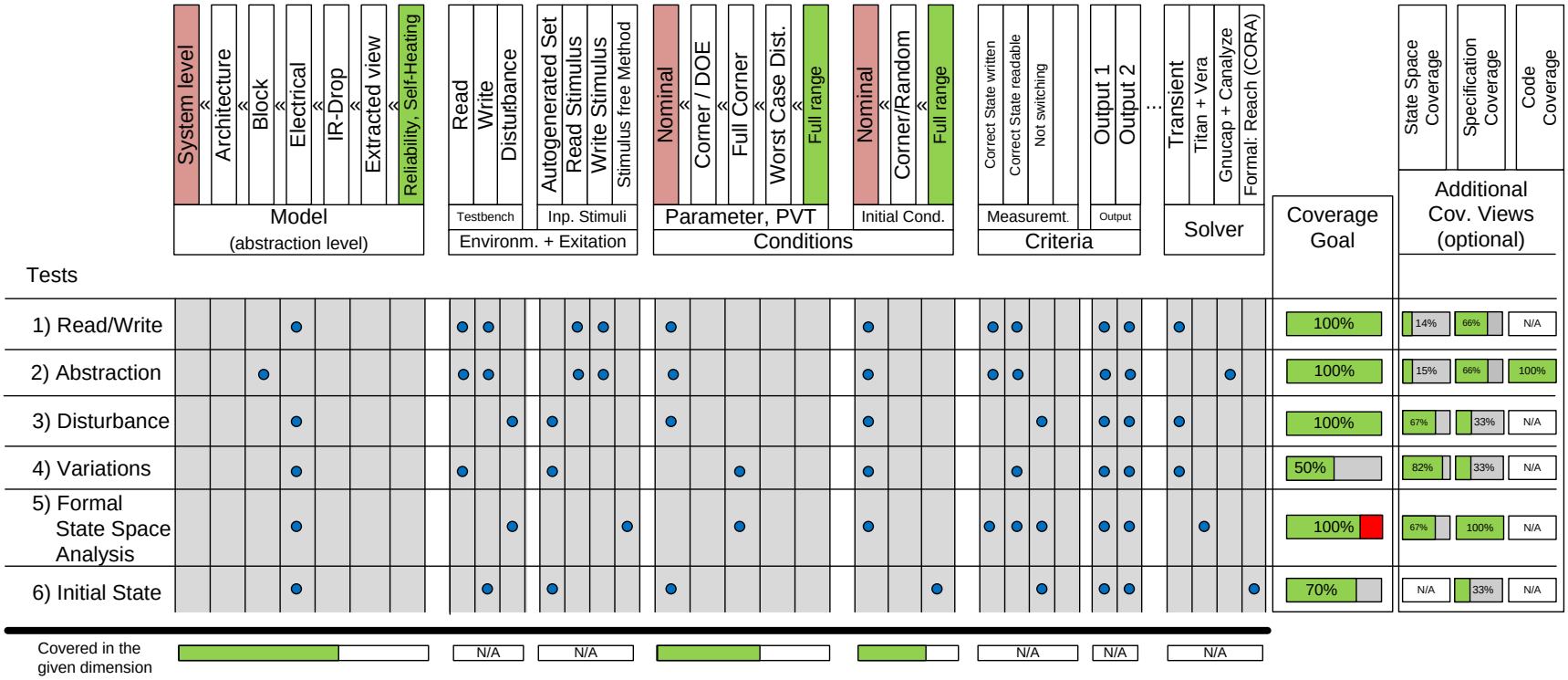
- State-Space-Coverage-Methods
  - Help finding reachable space
  - Help finding Counter examples for missing lines of code
  - Hard to compute
  - Best confidence
- IF State-Space Coverage is 100%:
  - Code-Coverage should be high
  - Except Parameter and Testbench variations
- Code-Coverage:
  - Simple to implement and use
  - Helps for TB, Model Signoff



Start with this
or similar mechanisms

COVA Tool Download:
http://www.em.cs.uni-frankfurt.de/index.php?id=218

What have we seen?



- Missing but available
  - Fault Coverage
  - Assertion Coverage
  - Specification Coverage
  - …

Tutorial A:

Analog Coverage – Yesterday‘s Dreams, Today‘s Reality and Tomorrow‘s Capabilities

Summary

Walter Hartong, Lars Hedrich, Markus Olbrich

Summary

- Main Ideas
  - Explicitly define coverage goal
  - Define verification tasks (simulations)
  - Measure coverage
  - Explicitly shrink coverage goal
- Many coverage measures used practically
- Advanced methods further increase coverage (including formal methods)

- First partial integration into commercial tools

Conclusion

- Future Challenges
  - Smooth integration of special coverages to be discussed
  - Convenient definition of verification goal

- Slides download
  - https://www.ims.uni-hannover.de/fileadmin/IMS/pdf/tutorialA2018.pdf

Thank you very much!